

APPENDIX

MSAvalon.Windows.Media

The following tables list the members exposed by the MSAvalon.Windows.Media namespace.

Classes

ArcSegment	Represents an elliptical arc between two points.
AudioData	Enables playing of audio files according to the state of a time node.
AudioDataConverter	AudioDataConverter
BezierSegment	Represents a cubic Bézier curve drawn between two points.
Brush	Provides a generic means for filling an area using solid colors (SolidColorBrush), gradients (LinearGradientBrush, RadialGradientBrush), images (ImageBrush), video, and more.
BrushConverter	Used to convert a Brush object to or from another object type.
Brushes	Implements a set of predefined solid colors.
CloseSegment	Represents a line that connects the last point of a PathFigure object with its starting point.
CodecFilter	Filter for enumerating codecs. Only those codecs that match the properties will be enumerated.
CodecInfo	Information about a specific codec and a factory for creating the codec. This is returned from the codec enumerator.
ColorCollection	
ColorCollectionConverter	ColorCollectionConverter - Converter class for converting instances of other types to and from ColorCollection instances.
ColorContext	
ColorConverter	Used to convert a Color object to or from another object type.
Colors	Implements a set of predefined colors.
ContainerVisual	Manages a collection of Visual objects.
DashArrays	DashArrays - The DashArrays class is static, and contains properties for well known dash styles.
DoubleCollection	
DoubleCollectionConverter	DoubleCollectionConverter - Converter class for converting instances of other types to and from DoubleCollection instances.
Drawing	A Drawing is a list of 2d drawing primitives.
DrawingBrush	DrawingBrush - This TileBrush defines its content as a Drawing
DrawingContext	Drawing context.
DrawingVisual	Visual that contains graphical content to be drawn.
EllipseGeometry	Represents the geometry of a circle or ellipse.
FontFamily	Font family
FormattedText	The FormattedText class is a part of Avalon MIL easy text API, which is targeted at programmers needing to add some simple text to a MIL visual.
Geometry	An abstract class that provides base functionality for all geometry classes, such as EllipseGeometry, RectangleGeometry, and PathGeometry. The Geometry class of objects can be used for clipping, hit-testing, and rendering 2-D graphic data.
GeometryCollection	Represents a collection of Geometry objects.
GetPageEventArgs	class GetPageEventArgs

GlyphRun	Glyph run class
GlyphTypeface	Physical font face corresponds to a font file on the disk
GradientBrush	An abstract class that describes a gradient fill. Classes that derive from GradientBrush describe different ways of interpreting gradient stops.
GradientStop	Describes the location and color of a transition point in a gradient.
GradientStopCollection	Represents a collection of GradientStop gradient stops.
HitTestParameters	This is the base class for packing together parameters for a hit test pass.
HitTestResult	This base returns the visual that was hit during a hit test pass.
HwndInterop	HwndInterop
HwndVisual	
HyphenationCandidate	Describes one Hyphenation candidate.
ICCPProfile	
ImageBrush	Fills an area with an image. This class may be used to specify images as the fill or background of other objects.
ImageCodecCollection	The collection of codecs (actually CodecInfos) on the system.
ImageCodecEnumerator	The enumerator for Image frames.
ImageColorTransform	ImageColorTransform Performs color management on an imaging pipeline.
ImageData	Contains an image and related data.
ImageDataBuilder	This object is used to build an ImageData object.
ImageDecoder	ImageDecoder is a container for image frames. Each image frame is an ImageSource. Unlike ImageSource, ImageDecoder is NOT an immutable object and can be re-initialized to a different image stream. However, any ImageSources (frames) that it returns must be immutable.
ImageDecoderBmp	The built-in Microsoft Bmp (Bitmap) Decoder.
ImageDecoderGif	The built-in Microsoft GIF Decoder.
ImageDecoderIcon	The built-in Microsoft Icon Decoder.
ImageDecoderInternal	For internal use only.
ImageDecoderJpeg	The built-in Microsoft Jpeg Decoder.
ImageDecoderPng	The built-in Microsoft PNG Decoder.
ImageDecoderTiff	The built-in Microsoft Tiff Decoder.
ImageEffect	The ImageEffect class is the base class for all imaging effects (blur, grayscale, etc) It's possible for an effect to not have any inputs but an effect must always have at least one output. The default implementations of things assume this. If a derived effect is going to play with Output/Outputs be sure that at least one is there.
ImageEffectBlur	Gaussian blur effect. It is a single input, single output effect. Warning: If the effect is being scaled (i.e. Input.ScaleX or Input.ScaleY isn't 1) and Expand is true, then it's possible for the output dimensions to be larger or smaller than PixelWidth and PixelHeight. Adjust the pixel buffer fed to copy to avoid problems.
ImageEffectFlipRotate	This effect can flip an image in X or Y and rotate by multiples of 90 deg
ImageEffectGammaCorrect	This effect changes the gamma of an image
ImageEffectGlow	Performs a glow effect. It is a single input, single output effect.
ImageEffectGrayscale	Converts an image to grayscale. It is a single input, single output effect.
ImageEffectNegate	Negates an image. It is a single input, single output effect.
ImageEffectSharpen	Unsharp mask. It is a single input, single output effect.
ImageEffectSource	ImageEffectSource class implementation
ImageEffectSourceCollection	The collection of image effect outputs

ImageEffectTint	Tint constructor. It is a single input, single output effect.
ImageEncoder	ImageEncoder collects a set of frames (ImageSource's) with their associated thumbnails and metadata and saves them to a specified stream. In addition to frame-specific thumbnails and metadata, there can also be an image-wide (global) thumbnail and metadata, if the codec supports it.
ImageEncoderBmp	Built-in Encoder for Bmp files.
ImageEncoderGif	Built-in Encoder for Gif files.
ImageEncoderInternal	ImageEncoderInternal collects a set of frames (ImageSource's) with their associated thumbnails and metadata and saves them to a specified stream. In addition to frame-specific thumbnails and metadata, there can also be an image-wide (global) thumbnail and metadata, if the codec supports it.
ImageEncoderJpeg	Built-in Encoder for Jpeg files.
ImageEncoderPng	Built-in Encoder for Png files.
ImageEncoderTiff	Built-in Encoder for Tiff files.
ImageExchangeMetaData	ImageExchangeMetaData This class is used to access and set metadata for ImageFiles which have Exif style metadata. MetaData is stored as Key/Value pairs, where Keys are not necessarily unique. This class provides generic access to all meta data within an image, as well as exposes CLR properties for certain well-known properties.
ImageExchangeProperty	ImageExchangeProperty - a tuple of an ImageExchangeID and the object which is the value of that property
ImageMetaData	ImageMetaData This class is used to access and set metadata for Images. This class also exposes a CodecMetaData property which exposes a codec-specific means of accessing the metadata for this image.
ImagePalette	ImagePalette class
ImageSizeOptions	Sizing options for an image. The resulting image will be scaled based on these options.
ImageSource	Defines the methods, properties, and events for the imaging pipeline, including decoders and effects.
ImageSourceCollection	The collection of codecs (actually ImageSource's) on the system.
ImageSourceConverter	ImageSourceConverter
IntegerCollection	
IntegerCollectionConverter	IntegerCollectionConverter - Converter class for converting instances of other types to and from IntegerCollection instances.
LinearGradientBrush	Defines a linear gradient used to fill an area.
LineGeometry	Represents the geometry of a line.
LineSegment	Represents a line between two points. Unlike LineGeometry objects, LineSegment must be contained within a PathFigure.
MatrixTransform	Creates an arbitrary affine matrix transformation used to manipulate objects or coordinate systems in a two-dimensional plane.
MediaData	MediaData. Use to playback Audio/Video content.
MediaSystem	The MediaSystem class controls the media layer.
NineGridBrush	Fills an entire area with an image. Portions of the image are stretched to fit within defined margins.
PathFigure	Represents a sub-section of a geometry, a single connected series of two-dimensional geometric segments.
PathFigureCollection	
PathFigureConverter	PathFigureConverter

PathGeometry	Represents a complex shape that may be composed of arcs, curves, ellipses, lines, and rectangles.
PathGeometryConverter	PathGeometryConverter
PathSegment	An abstract class that represents a segment of a PathFigure object. Classes that derive from PathSegment, such as ArcSegment, BezierSegment, and LineSegment, represent specific types of geometric segments.
PathSegmentCollection	Represents a list of PathSegment objects.
PathSegmentConverter	PathSegmentConverter
Pen	Describes how a shape is outlined.
PixelFormats	PixelFormats - The collection of supported Pixel Formats
PointCollection	
PointCollectionConverter	PointCollectionConverter - Converter class for converting instances of other types to and from PointCollection instances.
PointHitTestParameters	This is the class for specifying parameters hit testing with a point.
PointHitTestResult	This class returns the point and visual hit during a hit test pass.
PolyBezierSegment	PolyBezierSegment
PolyLineSegment	PolyLineSegment
PolyQuadraticBezierSegment	PolyQuadraticBezierSegment
PrintContext	PrintContext holds state and context for a printer interaction
QuadraticBezierSegment	QuadraticBezierSegment
RadialGradientBrush	Defines a radial gradient used to fill an object. A focal point defines the beginning of the gradient, and a circle defines the end point of the gradient.
RectangleGeometry	Represents the geometry of a rectangle.
RetainedVisual	RetainedVisual
RotateTransform	Used to rotate an object about a specified point in the two-dimensional x-y plane.
ScaleTransform	Scales an object in the two-dimensional x-y plane, starting from a defined center point. Scale factors are defined in x- and y-directions from this center point.
SkewTransform	Represents a two-dimensional skew.
SolidColorBrush	Represents a solid, uniform fill.
StartSegment	StartSegment
SubLineCollection	collection of subline. Subline can be object of one of these types GlyphRun LineOver Inline object
TileBrush	Abstract class that describes a way to fill a region with one or more "tiles." Derived classes define the different types of tiles that can be used; for example, the ImageBrush enables you to fill an area with an image.
Transform	An abstract class that you use as the parent class of all types of transformations in a two-dimensional plane, including rotation (RotateTransform), scale (ScaleTransform), skew (SkewTransform), and translation (TranslateTransform). This class hierarchy differs from the Matrix structure both because it is a class and because it supports animation and enumeration semantics.
TransformCollection	Used to create and manipulate a list of Transform objects.
TransformConverter	Used to convert a Transform object to or from another object type.
TranslateTransform	Translates an object in the two-dimensional x-y plane.
Typeface	A Typeface is a combination of family, weight, style and stretch:
VectorCollection	
VectorCollectionConverter	VectorCollectionConverter - Converter class for converting instances of other types to and from VectorCollection instances.

VideoData	Enables playing of video files according to the state of a time node.
VideoDataConverter	VideoDataConverter
Visual	Base class for all Visual types. It provides services and properties common to all Visuals, including hit-testing, coordinate transformation, and bounding box calculations.
VisualCollection	An ordered collection of Visual objects.
VisualManager	Renders a tree of Visual objects to a rendering target, typically a window.
Interfaces	
IHyphenate	IHyphenate is the interface for Hyphenation Service Provider If this interface is implemented on a class that is derived from a RetainedVisual, the RetainedVisual operations in validation mode, i.e. the graphics sub-system will call OnRender in a lazy fashion. (e.g. if the Visual appears for the first time on the screen). Note that OnRender can be called by the system anytime.
IRetainedRender	
IVisual	This interface defines the common methods and services available from a Visual object.
Enumerations	
BrushMappingMode	BrushMappingMode - Enum which describes whether certain values should be considered as absolute local coordinates or whether they should be considered multiples of a bounding box's size.
ChannelDescription	Describes order of each channel of pixel data
ColorInterpolationMode	ColorInterpolationMode - This determines how the colors in a gradient are interpolated.
CombineMode	Specifies the method used to combine two geometric areas.
FillRule	
GradientSpreadMethod	Specifies how the gradient should be drawn outside of the specified gradient vector or space.
HitTestFilterBehavior	Behavior for filtering visuals while hit testing
HitTestResultBehavior	Enum controls behavior when a positive hit occurs during hit testing.
HorizontalAlignment	The HorizontalAlignment enum is used to describe how content is positioned horizontally within a container.
HyphenationRule	Supported Hyphenation Rules.
ImagePaletteType	Pre-defined palette types
MediaState	Holds the current state of the Media
PenDashCap	PenDashCap - Enum which describes the drawing of the ends of a dash within a dashed line.
PenLineCap	Describes the shape at the end of a line or segment.
PenLineJoin	PenLineJoin - Enum which describes the drawing of the corners on the line.
Rotation	The rotation to be applied; only multiples of 90 degrees is supported.
StandardColorSpace	
Stretch	Stretch - Enum which describes how a source rect should be stretched to fit a destination rect.
StyleSimulations	Font style simulation
TiffCompressOptions	Compress options for saving TIFF image
TileMode	TileMode - Enum which describes the drawing of the ends of a line.
VerticalAlignment	The VerticalAlignment enum is used to describe how content is positioned vertically

within a container.

Structures

CharacterIndexer	This class is a helper to implement named indexers for characters.
Color	Represents colors in terms of alpha, red, green, and blue channels.
GlyphIndexer	This class is a helper to implement named indexers for glyph metrics.
ImageExchangeID	ImageExchangeID - This class is the type which can be used as the key for a property in an ImageMetaData instance. This can be either an integer or a string.
ImageExchangeMetaDataEnumerator	ImageExchangeMetaDataEnumerator The enumerator for ImageExchangeMetaData. Contains IEnumerator interface as well as strongly typed versions of the APIs
ImageFrameEnumerator	The enumerator for Image frames.
ImageMetaDataRational	An ImageMetaDataRational class is represented as a signed numerator and a signed denominator. The effective value of a rational is the numerator / denominator
ImageMetaDataUnsignedRational	A rational class is represented as an unsigned numerator and an unsigned denominator. The effective value of a rational is the numerator / denominator
ImagePaletteColor	ImagePaletteColor structure
IntegerRect	A rect composed of integer values. Typically used to specify the source rect (in pixels) of interest from an image.
Matrix	Represents a 3x3 matrix used for transformations in two-dimensional space. Because "Avalon" only allows affine transformations, the Matrix structure has six entries instead of nine.
NamedStringIndexer	This class is a helper to implement named indexers for strings localized in multiple cultures.
PixelFormat	Pixel Format Definition for images and pixel-based surfaces

Delegates

GetPageEventHandler	delegate GetPageEventHandler
HitTestFilterDelegate	Delegate for hit tester to control whether to test against children of visual.
HitTestResultDelegate	Delegate for hit tester to control returning of hit information on visual.

CLASSES**ArcSegment Class**

Definition: Represents an elliptical arc between two points.

Method	Description
ArcSegment	Initializes a new instance of the ArcSegment class.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Creates a copy of this ArcSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this ArcSegment that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from PathSegment.

IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
LargeArc	Gets or sets a Boolean that determines whether the arc is drawn with a sweep of 180 degrees or greater.
Point	Gets or sets the endpoint of the elliptical arc.
PointAnimations	Gets or sets a collection of PointModifier objects that animate the destination Point of the elliptical arc.
Size	Gets or sets The x- and y-radius of the elliptical arc.
SizeAnimations	Gets or sets a collection of SizeModifier objects that animate the size of the elliptical arc.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SweepFlag	Gets or sets a Boolean that determines whether the arc is drawn in a positive-angle or negative-angle direction.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
XRotation	Gets or sets a value that indicates how the ellipse is rotated relative to the current coordinate system.
XRotationAnimations	Gets or sets a collection of DoubleModifier objects that animate x-axis rotation factor of the elliptical arc.

Use a PathFigure object to store ArcSegment objects and other segments.

An elliptical arc is defined by its start and end points, x- and y-radius, x-axis rotation factor, a flag indicating how large the angle of the resulting arc should be, and another flag describing which direction the arc is drawn. The ArcSegment class does not contain a property for the starting point of the arc; it only defines the destination point of the arc it represents. The beginning point of the arc is the current point of the PathFigure to which the ArcSegment is added.

For most elliptical arcs of a particular position, size, and rotation, there are four different arcs that can be drawn; the LargeArc and SweepFlag properties indicate which arc to use.

Of the four candidate arc sweeps, two represent large arcs with sweeps of 180 degrees or greater, and two represent smaller arcs with sweeps 180 degrees or less. If LargeArc is true, then one of the two larger arc sweeps is chosen; otherwise, if LargeArc is false, one of the smaller arc sweeps is chosen.

If SweepFlag is true, the arc is drawn in a positive-angle direction. If SweepFlag is false, the arc is drawn in a negative-angle direction.

AudioData Class

Definition: Enables playing of audio files according to the state of a time node.

Method	Description
AudioData	Initializes a new instance of the AudioData class.
BeginIn	Schedules an interactive begin time. Inherited from MediaData.
CloneCore	Clones this MediaData.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from

MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.

	Changeable.
Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to <code>Enable</code> . Inherited from <code>MediaData</code> .
Dispose	Dispose the object. Inherited from <code>MediaData</code> .
EmbeddedChangeableReader	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
EmbeddedChangeableWriter	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
Enable	Enables this timeline, parenting it to the timeline specified by the <code>ParentTimeline</code> property. This allows the timeline to become active. This method throws an exception if the <code>ParentTimeline</code> property is null. Inherited from <code>MediaData</code> .
EndIn	Schedules an interactive end time. Inherited from <code>MediaData</code> .
Equals	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
Finalize	Finalizes the <code>MediaData</code> . Inherited from <code>MediaData</code> .
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
GetType	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
MakeUnchangeable	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
MakeUnchangeableCore	Implementation of <code>MakeUnchangeableCore</code> . Inherited from <code>MediaData</code> .
MemberwiseClone	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
ModifyHandlerIfChangeable	Adds or removes a <code>Changed</code> event handler to or from the specified <code>Changeable</code> object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
OnChanged	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
Pause	Pauses this media. Inherited from <code>MediaData</code> .
Play	Begins playback of media. Inherited from <code>MediaData</code> .
PropagateEventHandler	Propagates event handler to the timeline. Inherited from <code>MediaData</code> .
ReadPreamble	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
ReferenceEquals	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
Resume	Resumes this media. Inherited from <code>MediaData</code> .
Seek	Moves the timeline for this media. Inherited from <code>MediaData</code> .
ToString	Persist <code>MediaData</code> in a string. Inherited from <code>MediaData</code> .

ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Accesses the Acceleration SMIL attribute. Inherited from MediaData.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Accesses the AutoReverse SMIL attribute. Inherited from MediaData.
Begin	Accesses the Begin SMIL attribute. Inherited from MediaData.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	The current repetition iteration. Inherited from MediaData.
CurrentTime	The current time local to this media. Inherited from MediaData.
Deceleration	Accesses the Deceleration SMIL attribute. Inherited from MediaData.
Duration	Accesses the Duration SMIL attribute. Inherited from MediaData.
End	Accesses the End SMIL attribute. Inherited from MediaData.
EndSync	Accesses the EndSync SMIL attribute. Inherited from MediaData.
Fill	Accesses the Fill SMIL attribute. Inherited from MediaData.
FillDefault	Accesses the FillDefault SMIL attribute. Inherited from MediaData.
HasAudio	True if the media has audio output. Inherited from MediaData.
HasChanged	True if the media has changed since the last tick. Inherited from MediaData.
HasVideo	True if the media has a visual output. Inherited from MediaData.
Height	Get the video Height in pixels Inherited from MediaData.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	True if the media is active, false otherwise. Inherited from MediaData.
IsEnabled	True if the media is enabled, false otherwise. Inherited from MediaData.
IsForwardProgressing	True if the media is moving from past to future. Inherited from MediaData.
IsOverridingBaseValue	True if the media is either changing or in a fill state, false otherwise. Inherited from MediaData.
IsPaused	True if this media is paused. Inherited from MediaData.
IsReversed	True if this media is in a reverse period. Inherited from MediaData.
MediaDuration	Returns the native media duration. Inherited from MediaData.

Mute	Accesses the mute state of media playback. Inherited from MediaData.
ParentTimeline	Accesses the ParentTimeline attribute. Inherited from MediaData.
Progress	The current progress of the media, from 0 to 1. Inherited from MediaData.
RepeatCount	Accesses the RepeatCount SMIL attribute. Inherited from MediaData.
RepeatDuration	Accesses the RepeatDuration SMIL attribute. Inherited from MediaData.
Restart	Accesses the Restart SMIL attribute. Inherited from MediaData.
RestartDefault	Accesses the RestartDefault SMIL attribute. Inherited from MediaData.
Speed	Accesses the Speed SMIL attribute. Inherited from MediaData.
State	Returns the current state of the media. Inherited from MediaData.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Volume	Accesses the volume of media playback. Inherited from MediaData.
Width	Get the video Width in pixels Inherited from MediaData.

AudioDataConverter Class

Definition: AudioDataConverter

Method	Description
AudioDataConverter	
CanConvertFrom	Inherited from TypeConverter.
CanConvertFrom	CanConvertFrom
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	TypeConverter method override.
ConvertFrom	ConvertFromString
ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	TypeConverter method implementation.
ConvertTo	Inherited from TypeConverter.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.

GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

BezierSegment Class

Definition: Represents a cubic Bézier curve drawn between two points.

Method	Description
BezierSegment	Initializes a new instance of the BezierSegment class.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this BezierSegment.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.

GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this BezierSegment that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from PathSegment.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help

	determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Point1	Gets or sets the first control point of the curve.
Point1Animations	Gets or sets a collection of PointModifier objects that animate the first control point of the curve.
Point2	Gets or sets the second control point of the curve.
Point2Animations	Gets or sets a collection of PointModifier objects that animate the second control point of the curve.
Point3	Gets or sets the end point of the curve.
Point3Animations	Gets or sets a collection of PointModifier objects that animate the end point of the curve.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Use a PathFigure object to store BezierSegment objects and other segments. A cubic Bézier curve is defined by four points: a start point, an end point (Point3), and two control points (Point1 and Point2). The BezierSegment class does not contain a property for the starting point of the curve; it only defines the end point. The beginning point of the curve is the current point of the PathFigure to which the ArcSegment is added.

The two control points of a cubic Bézier curve behave like magnets, attracting portions of what would otherwise be a straight line towards themselves, producing a curve. The first control point, Point1, affects the beginning portion of the curve; the second control point, Point2, affects the ending portion of the curve. Note that the curve doesn't necessarily pass through either of the control points; each control point moves its portion of the line towards itself, but not through itself.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();

// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));

' VB .NET
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MS Avalon.Windows.Point(200,50))

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));

myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));

myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));

MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);

myPathFigure.AddSegment(myBezierSegment);

myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
myPathFigure.BezierTo(new MS Avalon.Windows.Point(400, 100), _
    new MS Avalon.Windows.Point(400, 200), new MS Avalon.Windows.Point(200, 300))

myPathFigure.BezierTo(new MS Avalon.Windows.Point(400, 300), _
    new MS Avalon.Windows.Point(400, 100), new MS Avalon.Windows.Point(200, 50))

myPathFigure.BezierTo(new MS Avalon.Windows.Point(0, 100), _
    new MS Avalon.Windows.Point(0, 200), new MS Avalon.Windows.Point(200,300))

Dim myBezierSegment As new BezierSegment( _
    new MS Avalon.Windows.Point(0, 300), new MS Avalon.Windows.Point(0, 100), _
    new MS Avalon.Windows.Point(200, 50), true)

myPathFigure.AddSegment(myBezierSegment)

' Add the PathFigure to the PathGeometry
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);
```

```

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

```

' VB .NET

```

Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

Dim myPath As new Path()
myPath.Data = myGeometryCollection

```

```

' Set the outline and the fill of the Path element.
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _
    MS Avalon.Windows.Media.Colors.Red)

```

```

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)

```

Geometry objects may also be rendered using the `DrawingContext`, which supplies a `DrawGeometry` method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

Brush Class

Definition: Provides a generic means for filling an area using solid colors (`SolidColorBrush`), gradients (`LinearGradientBrush`, `RadialGradientBrush`), images (`ImageBrush`), video, and more.

Method	Description
Brush	Initializes a new instance of the Brush class.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .

CloneCore	Subclasses must implement this to provide clones of themselves. Inherited from Animatable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	
GetIsOverridingBaseValue	
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.

PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer
SetDefaultParentTimeline	
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the brush has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
Opacity	Gets or sets the degree of opacity of a Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a

	DrawingContext command. Inherited from Changeable.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Use the Brushes class to fill an object using a predefined solid color, such as AliceBlue. A quick way to use a Brush in "Longhorn" markup language (code-named "XAML") is to supply the string for a color name from the Colors class, such as "Red" for the Red property or "ForestGreen" for the ForestGreen property. Hexadecimal notation is also valid.

This example creates simple vertical, horizontal, and radial gradients and uses them to fill an element using "XAML".

In the following example, vertical and horizontal gradients are used to set the Fill property of two Rectangle elements. In this particular example, the gradients are described using simple notation: GradientType StartColor EndColor, where GradientType is VerticalGradient, HorizontalGradient, or RadialGradient. StartColor and EndColor can be predefined color names (such as Blue) or hexadecimal values.

```
<Canvas xmlns="http://schemas.microsoft.com/2003/xaml">
```

```
  <Rectangle
    Fill="VerticalGradient Blue Green"
    RectangleLeft="20"
    RectangleTop="20"
    RectangleWidth="100"
    RectangleHeight="100">
  </Rectangle>
```

```
  <Rectangle
    Fill="HorizontalGradient Blue Red"
    RectangleLeft="120"
    RectangleTop="120"
    RectangleWidth="100"
    RectangleHeight="100">
  </Rectangle>
```

A vertical gradient is a linear gradient whose start and endpoints form a vertical line; likewise, a horizontal gradient is a linear gradient whose start and endpoints form a horizontal line. You can explicitly describe your own linear gradients using the following syntax:

LinearGradient StartPoint EndPoint StartColor EndColor, where StartPoint and EndPoint are the starting and ending coordinates, with each coordinate expressed as a pair of x and y values from 0 to 1, such as 0.1,0.1 and 0.5,0.5. These values indicate the relative position of the start or end point. An endpoint of 0.5,0.5 would be located 50 percent to the right of the fill area and 50 percent of the way from the top of the area—the middle of the shape.

In the following example, the Fill property of a Rectangle element is set by explicitly using a linear gradient.

```
<Rectangle
  Fill="LinearGradient 0.1,0.1 0.5,0.5 Blue Green"
  RectangleLeft="220"
  RectangleTop="220"
```

```

    RectangleWidth="100"
    RectangleHeight="100">
</Rectangle>

```

In the final example, the Fill property of a Rectangle element is set using a radial gradient.

```

<Rectangle
    Fill="RadialGradient Blue Red"
    RectangleLeft="320"
    RectangleTop="320"
    RectangleWidth="100"
    RectangleHeight="100">
</Rectangle>

```

```

</Canvas>

```

To create vertical and horizontal gradients in code or using compound notation, use the `LinearGradientBrush` class and set its `StartPoint` and `EndPoint` properties so that they describe a vertical or horizontal line. To create radial gradients in code or using compound notation, use the `RadialGradientBrush` class.

This example uses the `SolidColorBrush` class in "XAML" to color areas such as the window background and the border and interior of shapes. The `SolidColorBrush` creates fills that are uniform in color—it can't create gradient or pattern fills. `SolidColorBrush` uses both predefined color values and hexadecimal color values. In this example, several `Ellipse` shapes are created with identical fills and outlines, but the fills and outlines are specified in different formats to demonstrate the versatility of the `SolidColorBrush` class. In the following markup, a `Canvas` element is declared and its `Background` property is set to `LightGray`, one of the predefined colors.

```

<Canvas ID="root" xmlns="http://schemas.microsoft.com/2003/xaml"
    Background="LightGray">

```

In the next example, the `Fill` and `Stroke` properties of an `Ellipse` are set using ARGB notation. ARGB consists of a pound sign (#) and eight digits. The pound sign indicates that the digits that follow are in hexadecimal (base-16) format. The first two digits specify the alpha value, or opacity, of the color. FF indicates a color that is fully opaque, while 00 indicates a color that is completely transparent. The next six digits of the number specify the red, green, and blue values of the color.

The fill of the `Ellipse` is set to `#FFFFFF00`, which specifies a color that is fully opaque (FF), has the maximum amount of red (FF), the maximum amount of green (FF), and no blue(00). This combination produces yellow.

```

<Ellipse
    Fill="#FFFFFF00"
    CenterX="100"
    CenterY="200"
    RadiusX="75"
    RadiusY="75"
    StrokeThickness="5"
    Stroke="#FF0000FF"/>

```

In the next example, the `Fill` and `Stroke` properties of an `Ellipse` are set using shorter hexadecimal notation. The alpha value is omitted, and one digit is used for each red, green, and blue value instead of two. The resulting `Ellipse` has colors identical to the first.

```

<Ellipse
    Fill="#FF0"
    CenterX="200"

```

```

CenterY="200"
RadiusX="75"
RadiusY="75"
StrokeThickness="5"
Stroke="#00F"/>

```

In the next example, the Fill and Stroke properties of an Ellipse are set using two of the predefined colors, Yellow and Blue. The resulting Ellipse has colors identical to the previous two.

```

<Ellipse
  Fill="Yellow"
  CenterX="300"
  CenterY="200"
  RadiusX="75"
  RadiusY="75"
  StrokeThickness="5"
  Stroke="Blue"/>

```

In the final example, the Fill property of a Polyline is set by explicitly declaring a SolidColorBrush. The Color property of the SolidColorBrush is set to Blue, and its Opacity property is set to 0.4, creating a fill that is blue and 40 percent opaque (or 60 percent translucent).

```

<Polyline
  Points="300,200 400,125 400,275 300,200"
  Stroke="Purple"
  StrokeThickness="2.3">

  <Polyline.Fill>
    <SolidColorBrush Color="Blue" Opacity="0.4"/>
  </Polyline.Fill>

</Polyline>

</Canvas>

```

This example demonstrates several equivalent ways to specify color values to fill an area using code. The simplest syntax uses a named color property from the Brushes class. The Colors class provides the same named color properties which you can pass as an argument to SolidColorBrush. The example also shows how to pass Color values to SolidColorBrush as separate alpha, red, green, and blue values using the Color structure's static FromScRGB method. The example draws identical Ellipse shapes using identical colors specified in these different ways.

```

// C#
// Create the ellipses.
Ellipse e1 = new Ellipse();
Ellipse e2 = new Ellipse();
Ellipse e3 = new Ellipse();

// Set the fill value for the interior of each ellipse in
// different ways that have identical results.
e1.Fill = Brushes.Blue;
e2.Fill = new SolidColorBrush(Colors.Blue);
e3.Fill = new SolidColorBrush(Color.FromScRGB(1,0,0,1));

// Set the stroke value for the interior of each ellipse in
// different ways that have identical results.

```

```
e1.Stroke = Brushes.Black;
e2.Stroke = new SolidColorBrush(Colors.Black);
e3.Stroke = new SolidColorBrush(Color.FromScRGB(1,0,0,0));
```

This sample demonstrates only a few of the ways to instantiate Color objects. See Color for more information.

In order to actually render the ellipse the example also sets values for the StrokeThickness, CenterX, CenterY, RadiusX, and RadiusY properties in order to set the size and position of each ellipse.

```
// C#
// Set the thickness of the stroke.
e1.StrokeThickness = new Length(10);
e2.StrokeThickness = new Length(10);
e3.StrokeThickness = new Length(10);

// Set the size and position of the ellipses.
e1.CenterX = new Length(100);
e1.CenterY = new Length(75);
e1.RadiusX = new Length(50);
e1.RadiusY = new Length(50);

e2.CenterX = new Length(220);
e2.CenterY = new Length(75);
e2.RadiusX = new Length(50);
e2.RadiusY = new Length(50);

e3.CenterX = new Length(340);
e3.CenterY = new Length(75);
e3.RadiusX = new Length(50);
e3.RadiusY = new Length(50);
```

This example demonstrates how to make an element transparent or semi-transparent using the UIElement.Opacity and the Opacity property of Brush objects. The value of opacity properties is expressed as a value between 0 and 1, specifying a continuous range from fully transparent to fully opaque. A value of 0 specifies that the element is completely transparent while a value of 1 makes the element completely opaque. There are several ways of making an element transparent or semi-transparent; if the element has a background or a stroke property, you can set the Opacity property of the Brush associated with the stroke or background. Alternately, you can set the UIElement.Opacity property of the element, which affects the entire element and all its children.

In this example, three Ellipse elements, ellipses A, B, and C, are drawn in a Canvas and are made partially transparent. Ellipse A's UIElement.Opacity property is set to 0.5, making the entire element appear 50 percent opaque. Ellipse B's Fill is set explicitly using a SolidColorBrush. The SolidColorBrush object's Opacity property is set to 0.5. The resulting shape has a fully opaque Stroke and a 50 percent opaque Fill. Ellipse C has both its UIElement.Opacity property and the Opacity property of its SolidColorBrush set to 0.5, resulting in a shape with a 50 percent opaque Stroke and a 25 percent opaque Fill (0.5 multiplied by 0.5).

```
<Canvas xmlns="http://schemas.microsoft.com/2003/xaml"
Width="100%" Height="100%">
```

```
<!-- Omitted Code: A line is drawn behind the ellipses
and a fully opaque Ellipse is drawn for visual comparison.-->
```

```
<Ellipse
ID="A"
```

```

    Fill="Red"
    Stroke="Black"
    StrokeThickness="10"
    CenterX="140"
    CenterY="150"
    RadiusX="25"
    RadiusY="100"
    UIElement.Opacity="0.5"/>

```

```

<Text Canvas.Left="125" Canvas.Top="270" FontSize="20">A</Text>

```

```

<Ellipse
  ID="B"
  Stroke="Black"
  StrokeThickness="10"
  CenterX="210"
  CenterY="150"
  RadiusX="25"
  RadiusY="100">
  <Shape.Fill>
    <SolidColorBrush Color="Red" Opacity="0.5"/>
  </Shape.Fill>
</Ellipse>

```

```

<Text Canvas.Left="185" Canvas.Top="270" FontSize="20">B</Text>

```

```

<Ellipse
  ID="C"
  Stroke="Black"
  StrokeThickness="10"
  CenterX="280"
  CenterY="150"
  RadiusX="25"
  RadiusY="100"
  UIElement.Opacity="0.5">
  <Shape.Fill>
    <SolidColorBrush Color="Red" Opacity="0.5"/>
  </Shape.Fill>
</Ellipse>

```

```

<Text Canvas.Left="255" Canvas.Top="270" FontSize="20">C</Text>

```

```

</Canvas>

```

This example demonstrates how to create a gradient that has more than two colors in "XAML". To create a gradient with more than two colors, add a GradientStopCollection to the gradient's GradientStops property. Next, add GradientStop objects to the GradientStopCollection, one for each color the gradient should contain. Set the Color and the Offset, a value from 0 to 1 that determines the relative position of the stop in the gradient, of each of the stops. The following example shows a Button whose Background is filled with a horizontal gradient that has four colors.

```

<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

```

```

  <Button
    Canvas.Top="50"

```

```

Canvas.Left="50"
BorderBrush="Black"
Width="200"
Height="30">
<Button.Background>
<LinearGradientBrush >
<LinearGradientBrush.GradientStops>
<GradientStopCollection>
<GradientStop Color="Red" Offset="0" />
<GradientStop Color="Blue" Offset="0.25"/>
<GradientStop Color="Orange" Offset="0.75"/>
<GradientStop Color="Yellow" Offset="1"/>
</GradientStopCollection>
</LinearGradientBrush.GradientStops>
</LinearGradientBrush>
</Button.Background>
</Button>

</Canvas>

```

BrushConverter Class

Definition: Used to convert a Brush object to or from another object type.

Method	Description
BrushConverter	Initializes a new instance of the BrushConverter class.
CanConvertFrom	Determines whether this class can convert an object of a given type to a Brush object.
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	Determines whether this class can convert an object of a given type to the specified destination type.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	Converts from an object of a given type to a Brush object.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	Converts a Brush object to a specified type, using the specified context and culture information.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.

GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

Brushes Class

Definition: Implements a set of predefined solid colors.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
AliceBlue	Gets the solid fill color that has a hexadecimal value of #FFF0F8FF.
AntiqueWhite	Gets the solid fill color that has a hexadecimal value of #FFFAEBD7.
Aqua	Gets the solid fill color that has a hexadecimal value of #FF00FFFF.
Aquamarine	Gets the solid fill color that has a hexadecimal value of #FF7FFFD4.
Azure	Gets the solid fill color that has a hexadecimal value of #FFF0FFFF.
Beige	Gets the solid fill color that has a hexadecimal value of #FFF5F5DC.
Bisque	Gets the solid fill color that has a hexadecimal value of #FFFFE4C4.
Black	Gets the solid fill color that has a hexadecimal value of #FF000000.
BlanchedAlmond	Gets the solid fill color that has a hexadecimal value of #FFFEBBCD.
Blue	Gets the solid fill color that has a hexadecimal value of #FF0000FF.
BlueViolet	Gets the solid fill color that has a hexadecimal value of #FF8A2BE2.
Brown	Gets the solid fill color that has a hexadecimal value of #FFA52A2A.
BurlyWood	Gets the solid fill color that has a hexadecimal value of #FFDEB887.
CadetBlue	Gets the solid fill color that has a hexadecimal value of #FF5F9EA0.

Chartreuse	Gets the solid fill color that has a hexadecimal value of #FF7FFF00.
Chocolate	Gets the solid fill color that has a hexadecimal value of #FFD2691E.
Coral	Gets the solid fill color that has a hexadecimal value of #FFFF7F50.
CornflowerBlue	Gets the solid fill color that has a hexadecimal value of #FF6495ED.
Cornsilk	Gets the solid fill color that has a hexadecimal value of #FFFFF8DC.
Crimson	Gets the solid fill color that has a hexadecimal value of #FFDC143C.
Cyan	Gets the solid fill color that has a hexadecimal value of #FF00FFFF.
DarkBlue	Gets the solid fill color that has a hexadecimal value of #FF00008B.
DarkCyan	Gets the solid fill color that has a hexadecimal value of #FF008B8B.
DarkGoldenrod	Gets the solid fill color that has a hexadecimal value of #FFB8860B.
DarkGray	Gets the solid fill color that has a hexadecimal value of #FFA9A9A9.
DarkGreen	Gets the solid fill color that has a hexadecimal value of #FF006400.
DarkKhaki	Gets the solid fill color that has a hexadecimal value of #FFBDB76B.
DarkMagenta	Gets the solid fill color that has a hexadecimal value of #FF8B008B.
DarkOliveGreen	Gets the solid fill color that has a hexadecimal value of #FF556B2F.
DarkOrange	Gets the solid fill color that has a hexadecimal value of #FFFF8C00.
DarkOrchid	Gets the solid fill color that has a hexadecimal value of #FF9932CC.
DarkRed	Gets the solid fill color that has a hexadecimal value of #FF8B0000.
DarkSalmon	Gets the solid fill color that has a hexadecimal value of #FFE9967A.
DarkSeaGreen	Gets the solid fill color that has a hexadecimal value of #FF8FBC8B.
DarkSlateBlue	Gets the solid fill color that has a hexadecimal value of #FF483D8B.
DarkSlateGray	Gets the solid fill color that has a hexadecimal value of #FF2F4F4F.
DarkTurquoise	Gets the solid fill color that has a hexadecimal value of #FF00CED1.
DarkViolet	Gets the solid fill color that has a hexadecimal value of #FF9400D3.
DeepPink	Gets the solid fill color that has a hexadecimal value of #FFFF1493.
DeepSkyBlue	Gets the solid fill color that has a hexadecimal value of #FF00BFFF.
DimGray	Gets the solid fill color that has a hexadecimal value of #FF696969.
DodgerBlue	Gets the solid fill color that has a hexadecimal value of #FF1E90FF.
Firebrick	Gets the solid fill color that has a hexadecimal value of #FFB22222.
FloralWhite	Gets the solid fill color that has a hexadecimal value of #FFFFFFAF0.
ForestGreen	Gets the solid fill color that has a hexadecimal value of #FF228B22.
Fuchsia	Gets the solid fill color that has a hexadecimal value of #FFFF00FF.
Gainsboro	Gets the solid fill color that has a hexadecimal value of #FFDCDCDC.
GhostWhite	Gets the solid fill color that has a hexadecimal value of #FFF8F8FF.
Gold	Gets the solid fill color that has a hexadecimal value of #FFFFD700.
Goldenrod	Gets the solid fill color that has a hexadecimal value of #FFDAA520.
Gray	Gets the solid fill color that has a hexadecimal value of #FF808080.
Green	Gets the solid fill color that has a hexadecimal value of #FF008000.
GreenYellow	Gets the solid fill color that has a hexadecimal value of #FFADFF2F.
Honeydew	Gets the solid fill color that has a hexadecimal value of #FFF0FFF0.
HotPink	Gets the solid fill color that has a hexadecimal value of #FFFF69B4.
IndianRed	Gets the solid fill color that has a hexadecimal value of #FFCD5C5C.
Indigo	Gets the solid fill color that has a hexadecimal value of #FF4B0082.
Ivory	Gets the solid fill color that has a hexadecimal value of #FFFFFFF0.

Khaki	Gets the solid fill color that has a hexadecimal value of #FFF0E68C.
Lavender	Gets the solid fill color that has a hexadecimal value of #FFE6E6FA.
LavenderBlush	Gets the solid fill color that has a hexadecimal value of #FFFFFF0F5.
LawnGreen	Gets the solid fill color that has a hexadecimal value of #FF7CFC00.
LemonChiffon	Gets the solid fill color that has a hexadecimal value of #FFFFFFACD.
LightBlue	Gets the solid fill color that has a hexadecimal value of #FFADD8E6.
LightCoral	Gets the solid fill color that has a hexadecimal value of #FFF08080.
LightCyan	Gets the solid fill color that has a hexadecimal value of #FFE0FFFF.
LightGoldenrodYellow	Gets the solid fill color that has a hexadecimal value of #FFFAFAD2.
LightGray	Gets the solid fill color that has a hexadecimal value of #FFD3D3D3.
LightGreen	Gets the solid fill color that has a hexadecimal value of #FF90EE90.
LightPink	Gets the solid fill color that has a hexadecimal value of #FFFB6C1.
LightSalmon	Gets the solid fill color that has a hexadecimal value of #FFFA07A.
LightSeaGreen	Gets the solid fill color that has a hexadecimal value of #FF20B2AA.
LightSkyBlue	Gets the solid fill color that has a hexadecimal value of #FF87CEFA.
LightSlateGray	Gets the solid fill color that has a hexadecimal value of #FF778899.
LightSteelBlue	Gets the solid fill color that has a hexadecimal value of #FFB0C4DE.
LightYellow	Gets the solid fill color that has a hexadecimal value of #FFFFFFE0.
Lime	Gets the solid fill color that has a hexadecimal value of #FF00FF00.
LimeGreen	Gets the solid fill color that has a hexadecimal value of #FF32CD32.
Linen	Gets the solid fill color that has a hexadecimal value of #FFFAF0E6.
Magenta	Gets the solid fill color that has a hexadecimal value of #FFFF00FF.
Maroon	Gets the solid fill color that has a hexadecimal value of #FF800000.
MediumAquaMarine	Gets the solid fill color that has a hexadecimal value of #FF66CDAA.
MediumBlue	Gets the solid fill color that has a hexadecimal value of #FF0000CD.
MediumOrchid	Gets the solid fill color that has a hexadecimal value of #FFBA55D3.
MediumPurple	Gets the solid fill color that has a hexadecimal value of #FF9370DB.
MediumSeaGreen	Gets the solid fill color that has a hexadecimal value of #FF3CB371.
MediumSlateBlue	Gets the solid fill color that has a hexadecimal value of #FF7B68EE.
MediumSpringGreen	Gets the solid fill color that has a hexadecimal value of #FF00FA9A.
MediumTurquoise	Gets the solid fill color that has a hexadecimal value of #FF48D1CC.
MediumVioletRed	Gets the solid fill color that has a hexadecimal value of #FFC71585.
MidnightBlue	Gets the solid fill color that has a hexadecimal value of #FF191970.
MintCream	Gets the solid fill color that has a hexadecimal value of #FFF5FFFA.
MistyRose	Gets the solid fill color that has a hexadecimal value of #FFFFE4E1.
Moccasin	Gets the solid fill color that has a hexadecimal value of #FFFFE4B5.
NavajoWhite	Gets the solid fill color that has a hexadecimal value of #FFFDEAD.
Navy	Gets the solid fill color that has a hexadecimal value of #FF000080.
OldLace	Gets the solid fill color that has a hexadecimal value of #FFFD5E6.
Olive	Gets the solid fill color that has a hexadecimal value of #FF808000.
OliveDrab	Gets the solid fill color that has a hexadecimal value of #FF6B8E23.
Orange	Gets the solid fill color that has a hexadecimal value of #FFFA500.
OrangeRed	Gets the solid fill color that has a hexadecimal value of #FFFF4500.
Orchid	Gets the solid fill color that has a hexadecimal value of #FFDA70D6.

PaleGoldenrod	Gets the solid fill color that has a hexadecimal value of #FFEEE8AA.
PaleGreen	Gets the solid fill color that has a hexadecimal value of #FF98FB98.
PaleTurquoise	Gets the solid fill color that has a hexadecimal value of #FFAFEEEE.
PaleVioletRed	Gets the solid fill color that has a hexadecimal value of #FFDB7093.
PapayaWhip	Gets the solid fill color that has a hexadecimal value of #FFFFEFD5.
PeachPuff	Gets the solid fill color that has a hexadecimal value of #FFFDAB9.
Peru	Gets the solid fill color that has a hexadecimal value of #FFCD853F.
Pink	Gets the solid fill color that has a hexadecimal value of #FFFC0CB.
Plum	Gets the solid fill color that has a hexadecimal value of #FFDDA0DD.
PowderBlue	Gets the solid fill color that has a hexadecimal value of #FFB0E0E6.
Purple	Gets the solid fill color that has a hexadecimal value of #FF800080.
Red	Gets the solid fill color that has a hexadecimal value of #FFFF0000.
RosyBrown	Gets the solid fill color that has a hexadecimal value of #FFBC8F8F.
RoyalBlue	Gets the solid fill color that has a hexadecimal value of #FF4169E1.
SaddleBrown	Gets the solid fill color that has a hexadecimal value of #FF8B4513.
Salmon	Gets the solid fill color that has a hexadecimal value of #FFFA8072.
SandyBrown	Gets the solid fill color that has a hexadecimal value of #FFF4A460.
SeaGreen	Gets the solid fill color that has a hexadecimal value of #FF2E8B57.
SeaShell	Gets the solid fill color that has a hexadecimal value of #FFFFFF5EE.
Sienna	Gets the solid fill color that has a hexadecimal value of #FFA0522D.
Silver	Gets the solid fill color that has a hexadecimal value of #FFC0C0C0.
SkyBlue	Gets the solid fill color that has a hexadecimal value of #FF87CEEB.
SlateBlue	Gets the solid fill color that has a hexadecimal value of #FF6A5ACD.
SlateGray	Gets the solid fill color that has a hexadecimal value of #FF708090.
Snow	Gets the solid fill color that has a hexadecimal value of #FFFFFFAFA.
SpringGreen	Gets the solid fill color that has a hexadecimal value of #FF00FF7F.
SteelBlue	Gets the solid fill color that has a hexadecimal value of #FF4682B4.
Tan	Gets the solid fill color that has a hexadecimal value of #FFD2B48C.
Teal	Gets the solid fill color that has a hexadecimal value of #FF008080.
Thistle	Gets the solid fill color that has a hexadecimal value of #FFD8BFD8.
Tomato	Gets the solid fill color that has a hexadecimal value of #FFFF6347.
Transparent	Gets the solid fill color that has a hexadecimal value of #00FFFFFF.
Turquoise	Gets the solid fill color that has a hexadecimal value of #FF40E0D0.
Violet	Gets the solid fill color that has a hexadecimal value of #FFEE82EE.
Wheat	Gets the solid fill color that has a hexadecimal value of #FFF5DEB3.
White	Gets the solid fill color that has a hexadecimal value of #FFFFFFF.
WhiteSmoke	Gets the solid fill color that has a hexadecimal value of #FFF5F5F5.
Yellow	Gets the solid fill color that has a hexadecimal value of #FFFFF00.
YellowGreen	Gets the solid fill color that has a hexadecimal value of #FF9ACD32.

The "Avalon" color names match the Microsoft® .NET Framework version 1.0, Windows Forms, and Microsoft Internet Explorer color names. This representation is based on Unix X11 named color values. Color names are not case-sensitive.

The following color table image shows a color swatch for each of the named colors along with the corresponding color name, and hexadecimal value that you can use to render an area in that color.

This example uses the SolidColorBrush class in "Longhorn" markup language (code-named "XAML") to color areas such as the window background and the border and interior of shapes. The SolidColorBrush creates fills that are uniform in color—it can't create gradient or pattern fills. SolidColorBrush uses both predefined color values and hexadecimal color values. In this example, several Ellipse shapes are created with identical fills and outlines, but the fills and outlines are specified in different formats to demonstrate the versatility of the SolidColorBrush class.

In the following markup, a Canvas element is declared and its Background property is set to LightGray, one of the predefined colors.

```
<Canvas ID="root" xmlns="http://schemas.microsoft.com/2003/xaml"
  Background="LightGray">
```

In the next example, the Fill and Stroke properties of an Ellipse are set using ARGB notation. ARGB consists of a pound sign (#) and eight digits. The pound sign indicates that the digits that follow are in hexadecimal (base-16) format. The first two digits specify the alpha value, or opacity, of the color. FF indicates a color that is fully opaque, while 00 indicates a color that is completely transparent. The next six digits of the number specify the red, green, and blue values of the color.

The fill of the Ellipse is set to #FFFFFF00, which specifies a color that is fully opaque (FF), has the maximum amount of red (FF), the maximum amount of green (FF), and no blue(00). This combination produces yellow.

```
<Ellipse
  Fill="#FFFFFF00"
  CenterX="100"
  CenterY="200"
  RadiusX="75"
  RadiusY="75"
  StrokeThickness="5"
  Stroke="#FF0000FF"/>
```

In the next example, the Fill and Stroke properties of an Ellipse are set using shorter hexadecimal notation. The alpha value is omitted, and one digit is used for each red, green, and blue value instead of two. The resulting Ellipse has colors identical to the first.

```
<Ellipse
  Fill="#FF0"
  CenterX="200"
  CenterY="200"
  RadiusX="75"
  RadiusY="75"
  StrokeThickness="5"
  Stroke="#00F"/>
```

In the next example, the Fill and Stroke properties of an Ellipse are set using two of the predefined colors, Yellow and Blue. The resulting Ellipse has colors identical to the previous two.

```
<Ellipse
  Fill="Yellow"
  CenterX="300"
  CenterY="200"
  RadiusX="75"
  RadiusY="75"
  StrokeThickness="5"
  Stroke="Blue"/>
```

In the final example, the Fill property of a Polyline is set by explicitly declaring a SolidColorBrush. The Color property of the SolidColorBrush is set to Blue, and its Opacity property is set to 0.4, creating a fill that is blue and 40 percent opaque (or 60 percent translucent).

```
<Polyline
  Points="300,200 400,125 400,275 300,200"
  Stroke="Purple"
  StrokeThickness="2.3">

  <Polyline.Fill>
    <SolidColorBrush Color="Blue" Opacity="0.4"/>
  </Polyline.Fill>

</Polyline>

</Canvas>
```

This example demonstrates several equivalent ways to specify color values to fill an area using code. The simplest syntax uses a named color property from the Brushes class. The Colors class provides the same named color properties which you can pass as an argument to SolidColorBrush. The example also shows how to pass Color values to SolidColorBrush as separate alpha, red, green, and blue values using the Color structure's static FromScRGB method. The example draws identical Ellipse shapes using identical colors specified in these different ways.

```
// C#
// Create the ellipses.
Ellipse e1 = new Ellipse();
Ellipse e2 = new Ellipse();
Ellipse e3 = new Ellipse();

// Set the fill value for the interior of each ellipse in
// different ways that have identical results.
e1.Fill = Brushes.Blue;
e2.Fill = new SolidColorBrush(Colors.Blue);
e3.Fill = new SolidColorBrush(Color.FromScRGB(1,0,0,1));

// Set the stroke value for the interior of each ellipse in
// different ways that have identical results.
e1.Stroke = Brushes.Black;
e2.Stroke = new SolidColorBrush(Colors.Black);
e3.Stroke = new SolidColorBrush(Color.FromScRGB(1,0,0,0));
```

This sample demonstrates only a few of the ways to instantiate Color objects. See Color for more information.

In order to actually render the ellipse the example also sets values for the StrokeThickness, CenterX, CenterY, RadiusX, and RadiusY properties in order to set the size and position of each ellipse.

```
// C#
// Set the thickness of the stroke.
e1.StrokeThickness = new Length(10);
e2.StrokeThickness = new Length(10);
e3.StrokeThickness = new Length(10);

// Set the size and position of the ellipses.
```

```
e1.CenterX = new Length(100);
e1.CenterY = new Length(75);
e1.RadiusX = new Length(50);
e1.RadiusY = new Length(50);
```

```
e2.CenterX = new Length(220);
e2.CenterY = new Length(75);
e2.RadiusX = new Length(50);
e2.RadiusY = new Length(50);
```

```
e3.CenterX = new Length(340);
e3.CenterY = new Length(75);
e3.RadiusX = new Length(50);
e3.RadiusY = new Length(50);
```

The following code sample shows how to programatically change the Background color of a Border element. A Button element is placed near the center of a containing Canvas. The Canvas is nested within a Border element in order to display background and border properties. When the Button is clicked, the Brush color of the Border element is programatically changed to LightSteelBlue. In addition, Text is added to the Canvas indicating that the event has occurred. The text content of the Button element is also updated.

[C#]

```
<Border ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml"
  xmlns:def="Definition"
  BorderThickness="2"
  BorderBrush="black"
  Background="LightGray"
  Width="350"
  Height="350">

  <Canvas>

    <Button ID="btn" Canvas.Top="40" Canvas.Left="40" Background="LightSkyBlue" Width="100"
      Height="35" Click="ChangeBG">Click Me to change the Background Color</Button>
    <Text Canvas.Top="130" Canvas.Left="40" ID="Text1">Waiting for Click!</Text>

    <def:Code>
    <![CDATA[
      void ChangeBG(object sender, MS Avalon.Windows.Controls.ClickEventArgs e)
      {
        root.Background = MS Avalon.Windows.Media.Brushes.LightSteelBlue;
        btn.Content = "Clicked!";
        Text1.TextRange.Text = "The background is now LightSteelBlue";
      }
    ]]>
    </def:Code>

  </Canvas>

</Border>
```

The following code example shows how to complete the same operation in Microsoft Visual Basic® .NET.

[Visual Basic]

```
<Border ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml"
  xmlns:def="Definition"
    BorderThickness="2"
    BorderBrush="black"
    Background="LightGray"
    Width="350"
    Height="350">

  <Canvas>

    <Button ID="btn" Canvas.Top="40" Canvas.Left="40" Background="LightSkyBlue" Width="100"
Height="35"
Click="ChangeBG">Click Me to change the Background Color</Button>
    <Text Canvas.Top="130" Canvas.Left="40" ID="Text1">Waiting for Click!</Text>

    <def:Code>
    <![CDATA[
Sub ChangeBG(ByVal sender As Object, ByVal e As MS Avalon.Windows.Controls.ClickEventArgs)
    root.Background = MS Avalon.Windows.Media.Brushes.LightSteelBlue
    btn.Content = "Clicked!"
    Text1.TextRange.Text = "The background is now LightSteelBlue"

End Sub
]]>
    </def:Code>

  </Canvas>

</Border>
```

The following code example shows how to create a DockPanel in C#. Five Rectangle elements are instantiated and stacked within a parent DockPanel to partition space. Each rectangle is "Docked" within the parent DockPanel, with the final Rectangle set to "Fill" the remaining space.
[C#]

```
using System;
using MS Avalon.Windows;
using MS Avalon.Windows.Controls;
using MS Avalon.Windows.Media;

namespace Canvas_Demo
{
    public class MyApp : MS Avalon.Windows.Application
    {
        MS Avalon.Windows.Shapes.Rectangle rect1;
        MS Avalon.Windows.Shapes.Rectangle rect2;
        MS Avalon.Windows.Shapes.Rectangle rect3;
        MS Avalon.Windows.Shapes.Rectangle rect4;
        MS Avalon.Windows.Shapes.Rectangle rect5;
        MS Avalon.Windows.Controls.DockPanel dockPanel;
        MS Avalon.Windows.Window mainWindow;

        protected override void OnStartingUp (StartingUpCancelEventArgs e)
```

```

    {
        base.OnStartingUp (e);
        CreateAndShowMainWindow ();
    }

    private void CreateAndShowMainWindow ()
    {
        // Create the application's main window
        mainWindow = new MSAvalon.Windows.Window ();

        // Create a DockPanel with a width and height of 500 pixels
        dockPanel = new DockPanel ();
        dockPanel.Background = MSAvalon.Windows.Media.Brushes.Snow;
        mainWindow.Children.Add (dockPanel);
        dockPanel.Width = new Length (500);
        dockPanel.Height = new Length (500);

        // Add the first rectangle to the DockPanel
        rect1 = new MSAvalon.Windows.Shapes.Rectangle ();
        rect1.Stroke = MSAvalon.Windows.Media.Brushes.Black;
        rect1.Fill = MSAvalon.Windows.Media.Brushes.CadetBlue;
        rect1.Width = new Length (500);
        rect1.Height = new Length (25);
        MSAvalon.Windows.Controls.DockPanel.SetDock(rect1,
MSAvalon.Windows.Controls.Dock.Top);
        dockPanel.Children.Add (rect1);
        mainWindow.Show ();

        // Add the second rectangle to the DockPanel
        rect2 = new MSAvalon.Windows.Shapes.Rectangle ();
        rect2.Stroke = MSAvalon.Windows.Media.Brushes.Black;
        rect2.Fill = MSAvalon.Windows.Media.Brushes.LightSteelBlue;
        rect2.Width = new Length (500);
        rect2.Height = new Length (25);
        MSAvalon.Windows.Controls.DockPanel.SetDock(rect2,
MSAvalon.Windows.Controls.Dock.Top);
        dockPanel.Children.Add (rect2);
        mainWindow.Show ();

        // Add the third rectangle to the DockPanel
        rect4 = new MSAvalon.Windows.Shapes.Rectangle ();
        rect4.Stroke = MSAvalon.Windows.Media.Brushes.Black;
        rect4.Fill = MSAvalon.Windows.Media.Brushes.Teal;
        rect4.Width = new Length (500);
        rect4.Height = new Length (50);
        MSAvalon.Windows.Controls.DockPanel.SetDock(rect4,
MSAvalon.Windows.Controls.Dock.Bottom);
        dockPanel.Children.Add (rect4);
        mainWindow.Show ();

        // Add the fourth rectangle to the DockPanel
        rect3 = new MSAvalon.Windows.Shapes.Rectangle ();
        rect3.Stroke = MSAvalon.Windows.Media.Brushes.Black;
        rect3.Fill = MSAvalon.Windows.Media.Brushes.DarkSeaGreen;
        rect3.Width = new Length (200);
        rect3.Height = new Length (400);
    }

```

```

MSAvalon.Windows.Controls.DockPanel.SetDock(rect3,
MSAvalon.Windows.Controls.Dock.Left);
dockPanel.Children.Add (rect3);
mainWindow.Show ();

// Add the fifth rectangle to the DockPanel
rect5 = new MSAvalon.Windows.Shapes.Rectangle ();
rect5.Stroke = MSAvalon.Windows.Media.Brushes.Black;
rect5.Fill = MSAvalon.Windows.Media.Brushes.SlateGray;
MSAvalon.Windows.Controls.DockPanel.SetDock(rect5,
MSAvalon.Windows.Controls.Dock.Fill);
dockPanel.Children.Add (rect5);
mainWindow.Show ();
    }
}

internal sealed class EntryClass
{
    [System.STAThread()]
    private static void Main ()
    {
        MyApp app = new MyApp ();

        app.Run ();
    }
}

```

The following code example shows how to complete the same operation in Visual Basic .NET. [Visual Basic]

```

<DockPanel ID="myDP"
  xmlns="http://schemas.microsoft.com/2003/xaml"
  xmlns:def="Definition"
    Background="Snow"
    Loaded="onInit"
    Width="500"
    Height="500">
  <def:Code>
    <![CDATA[
Private Sub onInit(ByVal sender as object, ByVal args as System.EventArgs)

    ' Add the first Rectangle to the DockPanel
    Dim rect1 As new MSAvalon.Windows.Shapes.Rectangle
    rect1.Stroke = Brushes.Black
    rect1.Fill = Brushes.CadetBlue
    rect1.Width = new MSAvalon.Windows.Length(500)
    rect1.Height = new MSAvalon.Windows.Length(25)
    SetDock(rect1, MSAvalon.Windows.Controls.Dock.Top)
    myDP.Children.Add(rect1)

    ' Add the second Rectangle to the DockPanel
    Dim rect2 As new MSAvalon.Windows.Shapes.Rectangle
    rect2.Stroke = Brushes.Black
    rect2.Fill = Brushes.LightSteelBlue
    rect2.Width = new MSAvalon.Windows.Length(500)
    rect2.Height = new MSAvalon.Windows.Length(25)

```



```
SetDock(rect2, MSAvalon.Windows.Controls.Dock.Top)
myDP.Children.Add(rect2)
```

```
' Add the third Rectangle to the DockPanel
Dim rect3 As new MSAvalon.Windows.Shapes.Rectangle
rect3.Stroke = Brushes.Black
rect3.Fill = Brushes.Teal
rect3.Width = new MSAvalon.Windows.Length(500)
rect3.Height = new MSAvalon.Windows.Length(50)
SetDock(rect3, MSAvalon.Windows.Controls.Dock.Bottom)
myDP.Children.Add(rect3)
```

```
' Add the fourth Rectangle to the DockPanel
Dim rect4 As new MSAvalon.Windows.Shapes.Rectangle
rect4.Stroke = Brushes.Black
rect4.Fill = Brushes.DarkSeaGreen
rect4.Width = new MSAvalon.Windows.Length(200)
rect4.Height = new MSAvalon.Windows.Length(400)
SetDock(rect4, MSAvalon.Windows.Controls.Dock.Left)
myDP.Children.Add(rect4)
```

```
' Add the fourth Rectangle to the DockPanel
Dim rect5 As new MSAvalon.Windows.Shapes.Rectangle
rect5.Stroke = Brushes.Black
rect5.Fill = Brushes.SlateGray
SetDock(rect5, MSAvalon.Windows.Controls.Dock.Fill)
myDP.Children.Add(rect5)
```

```
End Sub
```

```
]]>
```

```
</def:Code>
```

```
</DockPanel>
```

CloseSegment Class

Definition: Represents a line that connects the last point of a PathFigure object with its starting point.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
CloseSegment	Initializes a new instance of the CloseSegment class.
Copy	Creates a copy of this CloseSegment.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.

DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this CloseSegment that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from PathSegment.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.

ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

CodecFilter Class

Definition: Filter for enumerating codecs. Only those codecs that match the properties will be enumerated.

Method	Description
CodecFilter	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.

GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
CodecAuthor	Who authored the codec.
FriendlyName	The Friendly name of a codec
HasDecoder	Find those codecs that have a matching decoder. Setting this to false means that we don't filter on whether or not there is a decoder for that codec.
HasEncoder	Find those codecs that have a matching encoder. Setting this to false means that we don't filter on whether or not there is an encoder for that codec.
ImageStream	Find a codec that can handle this image stream.
IsBuiltIn	Find those codecs that are built-in (not add-ins). Setting this to false means that we don't filter on whether or not the codec is one of the built-in ones.
MaxVersion	The maximum version number of the codec.
MimeType	Which Mime Types the codec supports.
MinVersion	The minimum version number of the codec.

CodecInfo Class

Definition: Information about a specific codec and a factory for creating the codec. This is returned from the codec enumerator.

Method	Description
CodecInfo	
CreateDecoderInstance	Get an instance of the decoder associated with this codec (if there is one).
CreateEncoderInstance	Get an instance of the encoder associated with this codec (if there is one).
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IsImageSupported	Whether the codec supports this image, based on looking at the first RequiredHeaderSize bytes from the image. The header must contain at least the first RequiredHeaderSize bytes from the image.
MatchesFilter	Note: this does NOT check/sniff the bytes in the stream.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Pr erty	Description
CodecAuthor	Who authored the codec.
CodecVersion	The version number of the codec.

FriendlyName	The Friendly name of a codec
HasDecoder	Whether there is a decoder associated with this codec.
HasEncoder	Whether there is an encoder associated with this codec.
IsBuiltIn	Whether this codec is one of the built-in ones.
MimeTypes	Which Mime Types the codec supports.
RequiredHeaderSize	The number of bytes needed from the image header to determine if the image is supported by this codec.

ColorCollection Class

Method	Description
Add	
AddRange	
Clear	
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorCollection	
Contains	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	
CopyTo	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	

IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.
Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.

CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	
Count	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ColorCollectionConverter Class

Definition: ColorCollectionConverter - Converter class for converting instances of other types to and from ColorCollection instances.

Method	Description
CanConvertFrom	Inherited from TypeConverter.
CanConvertFrom	CanConvertFrom - Returns whether or not this class can convert from a given type.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	CanConvertTo - Returns whether or not this class can convert to a given type.
ColorCollectionConverter	
ConvertFrom	ConvertFrom - Attempt to convert to a ColorCollection from the given object
ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	ConvertTo - Attempt to convert a ColorCollection to the given type
ConvertTo	Inherited from TypeConverter.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.

GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

ColorContext Class

Method	Description
ColorContext	ColorContext construct StandardColorSpace
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
ICCProfile	ICCProfile property

ColorCollectionConverter Class

Definition: ColorCollectionConverter - Converter class for converting instances of other types to and from ColorCollection instances.

Method	Description
CanConvertFrom	Inherited from TypeConverter.

CanConvertFrom	CanConvertFrom - Returns whether or not this class can convert from a given type.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	CanConvertTo - Returns whether or not this class can convert to a given type.
ColorCollectionConverter	
ConvertFrom	ConvertFrom - Attempt to convert to a ColorCollection from the given object
ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	ConvertTo - Attempt to convert a ColorCollection to the given type
ConvertTo	Inherited from TypeConverter.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

ColorContext Class

Method	Description
ColorContext	ColorContext construct StandardColorSpace
Equals	Determines whether two Object instances are equal. Inherited from Object.

Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
ICCProfile	ICCProfile property

ColorConverter Class

Definition: Used to convert a Color object to or from another object type.

Method	Description
CanConvertFrom	Inherited from TypeConverter.
CanConvertFrom	Determines whether an object of a given type can be converted to a Color object.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	Determines whether this class can convert an object of a given type to the specified destination type.
ColorConverter	Initializes a new instance of the ColorConverter class.
ConvertFrom	Converts from an object of a given type to a Color object.
ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Converts a string to a color.
ConvertTo	Converts a Color object to a specified type, using the specified context and culture information.
ConvertTo	Inherited from TypeConverter.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.

GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

Colors Class

Definition: Implements a set of predefined colors.

Method	Description
Colors	Instantiates the Colors class.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
AliceBlue	Gets the system-defined color that has an ARGB value of #FFF0F8FF.
AntiqueWhite	Gets the system-defined color that has an ARGB value of #FFFAEBD7.
Aqua	Gets the system-defined color that has an ARGB value of #FF00FFFF.
Aquamarine	Gets the system-defined color that has an ARGB value of #FF7FFFD4.
Azure	Gets the system-defined color that has an ARGB value of #FFF0FFFF.
Beige	Gets the system-defined color that has an ARGB value of #FFF5F5DC.
Bisque	Gets the system-defined color that has an ARGB value of #FFFFE4C4.
Black	Gets the system-defined color that has an ARGB value of #FF000000.
BlanchedAlmond	Gets the system-defined color that has an ARGB value of #FFFFEBCD.
Blue	Gets the system-defined color that has an ARGB value of #FF0000FF.
BlueViolet	Gets the system-defined color that has an ARGB value of #FF8A2BE2.
Brown	Gets the system-defined color that has an ARGB value of #FFA52A2A.
BurlyWood	Gets the system-defined color that has an ARGB value of #FFDEB887.

CadetBlue	Gets the system-defined color that has an ARGB value of #FF5F9EA0.
Chartreuse	Gets the system-defined color that has an ARGB value of #FF7FFF00.
Chocolate	Gets the system-defined color that has an ARGB value of #FFD2691E.
Coral	Gets the system-defined color that has an ARGB value of #FFFF7F50.
CornflowerBlue	Gets the system-defined color that has an ARGB value of #FF6495ED.
Cornsilk	Gets the system-defined color that has an ARGB value of #FFFFFF8DC.
Crimson	Gets the system-defined color that has an ARGB value of #FFDC143C.
Cyan	Gets the system-defined color that has an ARGB value of #FF00FFFF.
DarkBlue	Gets the system-defined color that has an ARGB value of #FF00008B.
DarkCyan	Gets the system-defined color that has an ARGB value of #FF008B8B.
DarkGoldenrod	Gets the system-defined color that has an ARGB value of #FFB8860B.
DarkGray	Gets the system-defined color that has an ARGB value of #FFA9A9A9.
DarkGreen	Gets the system-defined color that has an ARGB value of #FF006400.
DarkKhaki	Gets the system-defined color that has an ARGB value of #FFBDB76B.
DarkMagenta	Gets the system-defined color that has an ARGB value of #FF8B008B.
DarkOliveGreen	Gets the system-defined color that has an ARGB value of #FF556B2F.
DarkOrange	Gets the system-defined color that has an ARGB value of #FFFF8C00.
DarkOrchid	Gets the system-defined color that has an ARGB value of #FF9932CC.
DarkRed	Gets the system-defined color that has an ARGB value of #FF8B0000.
DarkSalmon	Gets the system-defined color that has an ARGB value of #FFE9967A.
DarkSeaGreen	Gets the system-defined color that has an ARGB value of #FF8FBC8B.
DarkSlateBlue	Gets the system-defined color that has an ARGB value of #FF483D8B.
DarkSlateGray	Gets the system-defined color that has an ARGB value of #FF2F4F4F.
DarkTurquoise	Gets the system-defined color that has an ARGB value of #FF00CED1.
DarkViolet	Gets the system-defined color that has an ARGB value of #FF9400D3.
DeepPink	Gets the system-defined color that has an ARGB value of #FFFF1493.
DeepSkyBlue	Gets the system-defined color that has an ARGB value of #FF00BFFF.
DimGray	Gets the system-defined color that has an ARGB value of #FF696969.
DodgerBlue	Gets the system-defined color that has an ARGB value of #FF1E90FF.
Firebrick	Gets the system-defined color that has an ARGB value of #FF451A22.
FloralWhite	Gets the system-defined color that has an ARGB value of #FFFFFFAF0.
ForestGreen	Gets the system-defined color that has an ARGB value of #FF228B22.
Fuchsia	Gets the system-defined color that has an ARGB value of #FFFF00FF.
Gainsboro	Gets the system-defined color that has an ARGB value of #FFDCDCDC.
GhostWhite	Gets the system-defined color that has an ARGB value of #FFF8F8FF.
Gold	Gets the system-defined color that has an ARGB value of #FFD700.
Goldenrod	Gets the system-defined color that has an ARGB value of #FFDAA520.
Gray	Gets the system-defined color that has an ARGB value of #FF808080.
Green	Gets the system-defined color that has an ARGB value of #FF008000.
GreenYellow	Gets the system-defined color that has an ARGB value of #FFADFF2F.
Honeydew	Gets the system-defined color that has an ARGB value of #FFF0FFF0.
HotPink	Gets the system-defined color that has an ARGB value of #FFFF69B4.
IndianRed	Gets the system-defined color that has an ARGB value of #FFCD5C5C.
Indigo	Gets the system-defined color that has an ARGB value of #FF4B0082.

Ivory	Gets the system-defined color that has an ARGB value of #FFFFFFF0.
Khaki	Gets the system-defined color that has an ARGB value of #FFF0E68C.
Lavender	Gets the system-defined color that has an ARGB value of #FFE6E6FA.
LavenderBlush	Gets the system-defined color that has an ARGB value of #FFFFFF0F5.
LawnGreen	Gets the system-defined color that has an ARGB value of #FF7CFC00.
LemonChiffon	Gets the system-defined color that has an ARGB value of #FFFFFFACD.
LightBlue	Gets the system-defined color that has an ARGB value of #FFADD8E6.
LightCoral	Gets the system-defined color that has an ARGB value of #FFF08080.
LightCyan	Gets the system-defined color that has an ARGB value of #FFE0FFFF.
LightGoldenrodYellow	Gets the system-defined color that has an ARGB value of #FFFAFAD2.
LightGray	Gets the system-defined color that has an ARGB value of #FFD3D3D3.
LightGreen	Gets the system-defined color that has an ARGB value of #FF90EE90.
LightPink	Gets the system-defined color that has an ARGB value of #FFFB6C1.
LightSalmon	Gets the system-defined color that has an ARGB value of #FFFA07A.
LightSeaGreen	Gets the system-defined color that has an ARGB value of #FF20B2AA.
LightSkyBlue	Gets the system-defined color that has an ARGB value of #FF87CEFA.
LightSlateGray	Gets the system-defined color that has an ARGB value of #FF778899.
LightSteelBlue	Gets the system-defined color that has an ARGB value of #FFB0C4DE.
LightYellow	Gets the system-defined color that has an ARGB value of #FFFFFFFE0.
Lime	Gets the system-defined color that has an ARGB value of #FF00FF00.
LimeGreen	Gets the system-defined color that has an ARGB value of #FF32CD32.
Linen	Gets the system-defined color that has an ARGB value of #FFFAF0E6.
Magenta	Gets the system-defined color that has an ARGB value of #FFFF00FF.
Maroon	Gets the system-defined color that has an ARGB value of #FF800000.
MediumAquamarine	Gets the system-defined color that has an ARGB value of #FF66CDAA.
MediumBlue	Gets the system-defined color that has an ARGB value of #FF0000CD.
MediumOrchid	Gets the system-defined color that has an ARGB value of #FFBA55D3.
MediumPurple	Gets the system-defined color that has an ARGB value of #FF9370DB.
MediumSeaGreen	Gets the system-defined color that has an ARGB value of #FF3CB371.
MediumSlateBlue	Gets the system-defined color that has an ARGB value of #FF7B68EE.
MediumSpringGreen	Gets the system-defined color that has an ARGB value of #FF00FA9A.
MediumTurquoise	Gets the system-defined color that has an ARGB value of #FF48D1CC.
MediumVioletRed	Gets the system-defined color that has an ARGB value of #FFC71585.
MidnightBlue	Gets the system-defined color that has an ARGB value of #FF191970.
MintCream	Gets the system-defined color that has an ARGB value of #FFF5FFFA.
MistyRose	Gets the system-defined color that has an ARGB value of #FFFFE4E1.
Moccasin	Gets the system-defined color that has an ARGB value of #FFFFE4B5.
NavajoWhite	Gets the system-defined color that has an ARGB value of #FFFFDEAD.
Navy	Gets the system-defined color that has an ARGB value of #FF000080.
OldLace	Gets the system-defined color that has an ARGB value of #FFFD5E6.
Olive	Gets the system-defined color that has an ARGB value of #FF808000.
OliveDrab	Gets the system-defined color that has an ARGB value of #FF6B8E23.
Orange	Gets the system-defined color that has an ARGB value of #FFFA500.
OrangeRed	Gets the system-defined color that has an ARGB value of #FFFF4500.

Orchid	Gets the system-defined color that has an ARGB value of #FFDA70D6.
PaleGoldenrod	Gets the system-defined color that has an ARGB value of #FFEEE8AA.
PaleGreen	Gets the system-defined color that has an ARGB value of #FF98FB98.
PaleTurquoise	Gets the system-defined color that has an ARGB value of #FFAFEEEE.
PaleVioletRed	Gets the system-defined color that has an ARGB value of #FFDB7093.
PapayaWhip	Gets the system-defined color that has an ARGB value of #FFFFEFD5.
PeachPuff	Gets the system-defined color that has an ARGB value of #FFFFDAB9.
Peru	Gets the system-defined color that has an ARGB value of #FFCD853F.
Pink	Gets the system-defined color that has an ARGB value of #FFFC0CB.
Plum	Gets the system-defined color that has an ARGB value of #FFDDA0DD.
PowderBlue	Gets the system-defined color that has an ARGB value of #FFB0E0E6.
Purple	Gets the system-defined color that has an ARGB value of #FF800080.
Red	Gets the system-defined color that has an ARGB value of #FFFF0000.
RosyBrown	Gets the system-defined color that has an ARGB value of #FFBC8F8F.
RoyalBlue	Gets the system-defined color that has an ARGB value of #FF4169E1.
SaddleBrown	Gets the system-defined color that has an ARGB value of #FF8B4513.
Salmon	Gets the system-defined color that has an ARGB value of #FFFA8072.
SandyBrown	Gets the system-defined color that has an ARGB value of #FFF4A460.
SeaGreen	Gets the system-defined color that has an ARGB value of #FF2E8B57.
SeaShell	Gets the system-defined color that has an ARGB value of #FFFFFF5EE.
Sienna	Gets the system-defined color that has an ARGB value of #FFA0522D.
Silver	Gets the system-defined color that has an ARGB value of #FFC0C0C0.
SkyBlue	Gets the system-defined color that has an ARGB value of #FF87CEEB.
SlateBlue	Gets the system-defined color that has an ARGB value of #FF6A5ACD.
SlateGray	Gets the system-defined color that has an ARGB value of #FF708090.
Snow	Gets the system-defined color that has an ARGB value of #FFFFFFAFA.
SpringGreen	Gets the system-defined color that has an ARGB value of #FF00FF7F.
SteelBlue	Gets the system-defined color that has an ARGB value of #FF4682B4.
Tan	Gets the system-defined color that has an ARGB value of #FFD2B48C.
Teal	Gets the system-defined color that has an ARGB value of #FF008080.
Thistle	Gets the system-defined color that has an ARGB value of #FFD8BFD8.
Tomato	Gets the system-defined color that has an ARGB value of #FFFF6347.
Transparent	Gets the system-defined color that has an ARGB value of #00FFFFFF.
Turquoise	Gets the system-defined color that has an ARGB value of #FF40E0D0.
Violet	Gets the system-defined color that has an ARGB value of #FFEE82EE.
Wheat	Gets the system-defined color that has an ARGB value of #FFF5DEB3.
White	Gets the system-defined color that has an ARGB value of #FFFFFFFF.
WhiteSmoke	Gets the system-defined color that has an ARGB value of #FFF5F5F5.
Yellow	Gets the system-defined color that has an ARGB value of #FFFFF00.
YellowGreen	Gets the system-defined color that has an ARGB value of #FF9ACD32.

The "Avalon" color names match the Microsoft® .NET Framework version 1.0, Windows Forms, and Microsoft Internet Explorer color names. This representation is based on UNIX X11 named color values. Color names are not case-sensitive.

The following color table image shows a color swatch for each of the named colors along with the corresponding color name, hexadecimal value, and RGB value that you can use to render an area in that color.

This example uses the SolidColorBrush class in "Longhorn" markup language (code-named "XAML") to color areas such as the window background and the border and interior of shapes. The SolidColorBrush creates fills that are uniform in color—it can't create gradient or pattern fills. SolidColorBrush uses both predefined color values and hexadecimal color values. In this example, several Ellipse shapes are created with identical fills and outlines, but the fills and outlines are specified in different formats to demonstrate the versatility of the SolidColorBrush class.

In the following markup, a Canvas element is declared and its Background property is set to LightGray, one of the predefined colors.

```
<Canvas ID="root" xmlns="http://schemas.microsoft.com/2003/xaml"
  Background="LightGray">
```

In the next example, the Fill and Stroke properties of an Ellipse are set using ARGB notation. ARGB consists of a pound sign (#) and eight digits. The pound sign indicates that the digits that follow are in hexadecimal (base-16) format. The first two digits specify the alpha value, or opacity, of the color. FF indicates a color that is fully opaque, while 00 indicates a color that is completely transparent. The next six digits of the number specify the red, green, and blue values of the color.

The fill of the Ellipse is set to #FFFFFF00, which specifies a color that is fully opaque (FF), has the maximum amount of red (FF), the maximum amount of green (FF), and no blue(00). This combination produces yellow.

```
<Ellipse
  Fill="#FFFFFF00"
  CenterX="100"
  CenterY="200"
  RadiusX="75"
  RadiusY="75"
  StrokeThickness="5"
  Stroke="#FF0000FF"/>
```

In the next example, the Fill and Stroke properties of an Ellipse are set using shorter hexadecimal notation. The alpha value is omitted, and one digit is used for each red, green, and blue value instead of two. The resulting Ellipse has colors identical to the first.

```
<Ellipse
  Fill="#FF0"
  CenterX="200"
  CenterY="200"
  RadiusX="75"
  RadiusY="75"
  StrokeThickness="5"
  Stroke="#00F"/>
```

In the next example, the Fill and Stroke properties of an Ellipse are set using two of the predefined colors, Yellow and Blue. The resulting Ellipse has colors identical to the previous two.

```
<Ellipse
  Fill="Yellow"
  CenterX="300"
  CenterY="200"
```

```

RadiusX="75"
RadiusY="75"
StrokeThickness="5"
Stroke="Blue"/>

```

In the final example, the Fill property of a Polyline is set by explicitly declaring a SolidColorBrush. The Color property of the SolidColorBrush is set to Blue, and its Opacity property is set to 0.4, creating a fill that is blue and 40 percent opaque (or 60 percent translucent).

```

<Polyline
  Points="300,200 400,125 400,275 300,200"
  Stroke="Purple"
  StrokeThickness="2.3">

  <Polyline.Fill>
    <SolidColorBrush Color="Blue" Opacity="0.4"/>
  </Polyline.Fill>

</Polyline>

</Canvas>

```

This example demonstrates several equivalent ways to specify color values to fill an area using code. The simplest syntax uses a named color property from the Brushes class. The Colors class provides the same named color properties which you can pass as an argument to SolidColorBrush. The example also shows how to pass Color values to SolidColorBrush as separate alpha, red, green, and blue values using the Color structure's static FromScRGB method. The example draws identical Ellipse shapes using identical colors specified in these different ways.

```

// C#
// Create the ellipses.
Ellipse e1 = new Ellipse();
Ellipse e2 = new Ellipse();
Ellipse e3 = new Ellipse();

// Set the fill value for the interior of each ellipse in
// different ways that have identical results.
e1.Fill = Brushes.Blue;
e2.Fill = new SolidColorBrush(Colors.Blue);
e3.Fill = new SolidColorBrush(Color.FromScRGB(1,0,0,1));

// Set the stroke value for the interior of each ellipse in
// different ways that have identical results.
e1.Stroke = Brushes.Black;
e2.Stroke = new SolidColorBrush(Colors.Black);
e3.Stroke = new SolidColorBrush(Color.FromScRGB(1,0,0,0));

```

This sample demonstrates only a few of the ways to instantiate Color objects. See Color for more information.

In order to actually render the ellipse the example also sets values for the StrokeThickness, CenterX, CenterY, RadiusX, and RadiusY properties in order to set the size and position of each ellipse.

```

// C#
// Set the thickness of the stroke.
e1.StrokeThickness = new Length(10);

```



```

e2.StrokeThickness = new Length(10);
e3.StrokeThickness = new Length(10);

// Set the size and position of the ellipses.
e1.CenterX = new Length(100);
e1.CenterY = new Length(75);
e1.RadiusX = new Length(50);
e1.RadiusY = new Length(50);

e2.CenterX = new Length(220);
e2.CenterY = new Length(75);
e2.RadiusX = new Length(50);
e2.RadiusY = new Length(50);

e3.CenterX = new Length(340);
e3.CenterY = new Length(75);
e3.RadiusX = new Length(50);
e3.RadiusY = new Length(50);

```

ContainerVisual Class

Definition: Manages a collection of Visual objects.

Method	Description
ClearValue	Clears the local value of a property Inherited from DependencyObject.
ContainerVisual	Creates a new instance of the ContainerVisual class.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Releases all resources held by the Visual object. Inherited from Visual.
FindCommonVisualAncestor	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetLocalValueEnumerator	Create a local value enumerator for this instance Inherited from DependencyObject.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Retrieve the value of a property Inherited from DependencyObject.
HitTest	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
HitTestCore	HitTestCore implements whether we have hit the bounds of this visual. Inherited from Visual.
InvalidateProperty	Invalidates a property Inherited from DependencyObject.
IsAncestorOf	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
IsDescendantOf	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
IVisual.FindCommonVisualAncestor	Inherited from Visual.
IVisual.HitTest	Inherited from Visual.

IVisual.IsAncestorOf	Inherited from Visual.
IVisual.IsDescendantOf	Inherited from Visual.
IVisual.TransformFromAncestor	Inherited from Visual.
IVisual.TransformFromDescendant	Inherited from Visual.
IVisual.TransformFromVisual	Inherited from Visual.
IVisual.TransformToAncestor	Inherited from Visual.
IVisual.TransformToDescendant	Inherited from Visual.
IVisual.TransformToVisual	Inherited from Visual.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
OnDelayedInvalidate	TODO: Left over from WCP FastBuild, determine relevance in future version of FastBuild Inherited from DependencyObject.
OnPropertyInvalidated	Notification that a specified property has been invalidated Inherited from DependencyObject.
ReadLocalValue	Retrieve the local value of a property (if set) Inherited from DependencyObject.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
SetValue	Sets the local value of a property Inherited from DependencyObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
TransformFromAncestor	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
TransformFromDescendant	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
TransformFromVisual	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
TransformToAncestor	Returns a Matrix object that represents the aggregate transformation from the coordinates of a Visual to the specified ancestor.
TransformToDescendant	Returns a transformation matrix that can be used to transform coordinates from this node to a specified descendant Visual.
TransformToVisual	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
ValidateProperty	Retrieve the value of a property (for use by native cache backed custom get accessors) Inherited from DependencyObject.
ValidatePropertyCore	Allows subclasses to participate in property value computation Inherited from DependencyObject.

Property	Description
Children	Gets a collection of the ContainerVisual's children.
Clip	Gets or sets the clipping region of this ContainerVisual.
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
DependencyObjectType	Returns the DType that represents the CLR type of this instance Inherited from DependencyObject.

HasChildren	Gets a value that indicates whether the ContainerVisual has a child collection.
HitTestBounds	HitBounds returns the hit region bounding box for the current visual. Inherited from Visual.
IsDisposed	Gets a value that indicates whether the system has disposed of the Visual. Inherited from Visual.
Opacity	Gets or sets the opacity of the ContainerVisual.
Parent	Gets the parent Visual.

DashArrays Class

Definition: DashArrays - The DashArrays class is static, and contains properties for well known dash styles.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Dash	Dash - A DashArray which is 3 on, 1 off
DashDot	DashDot - A DashArray which is 3 on, 1 off, 1 on, 1 off
DashDotDot	DashDot - A DashArray which is 3 on, 1 off, 1 on, 1 off, 1 on, 1 off
Dot	Dot - A DashArray which is 1 on, 1 off
Solid	Solid - A solid DashArray (no dashes).

DoubleCollection Class

Method	Description
Add	
AddRange	
Clear	
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	
CopyTo	Copies the entire DoubleCollection to a compatible one-dimensional

	Array, starting at the specified index of the target array.
DoubleCollection	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.

ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	
Count	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

DoubleCollectionConverter Class

Definition: DoubleCollectionConverter - Converter class for converting instances of other types to and from DoubleCollection instances.

Method	Description
CanConvertFrom	CanConvertFrom - Returns whether or not this class can convert from a given type.
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.

CanConvertTo	CanConvertTo - Returns whether or not this class can convert to a given type.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	ConvertFrom - Attempt to convert to a DoubleCollection from the given object
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	ConvertTo - Attempt to convert a DoubleCollection to the given type
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
DoubleCollectionConverter	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

Drawing Class

Definition: A Drawing is a list of 2d drawing primitives.

Method	Description
CloneCore	CloneCore - the internal implementation of the Changeable clone.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.

Copy	Make a copy of this Changeable
Drawing	Constructor for Drawing. This initializes to empty.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Finalizer for Drawing
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore - the internal implementation of the code which makes this instance immutable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
Open	Opens the Drawing for rendering. The returned DrawingContext can be used to render into the Drawing.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
Bounds	Bounds Property - The bounds of the contents.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

DrawingBrush Class

Definition: DrawingBrush - This TileBrush defines its content as a Drawing.

Method	Description
CloneCore	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this TileBrush. Inherited from TileBrush.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this DrawingBrush.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	Inherited from Brush.
DrawingBrush	Default constructor for DrawingBrush. The resulting Brush has no content.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.

EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Brush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this DrawingBrush that represents its current state.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this TileBrush that represents its current state. Inherited from TileBrush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.

SetDefaultParentTimelineCore	Inherited from Brush.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
ContentUnits	Gets or sets a BrushMappingMode enumeration that specifies whether the value of the brush's ViewBox—the size and position of the brush's content—is relative to the size of the output area. This property only has an effect when the size of the brush's ViewPort is set to Rect.Empty. Inherited from TileBrush.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
Drawing	Drawing - Read Only accessor for the Drawing which describes the contents of this brush. Default is null.
HasAnimations	Gets a Boolean that indicates whether the brush has animations. Inherited from TileBrush.
HorizontalAlignment	Gets or sets a HorizontalAlignment enumeration that specifies how the brush's content is horizontally aligned within its tiles. Inherited from TileBrush.
IsAnimating	Inherited from TileBrush.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Inherited from TileBrush.
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

Stretch	Gets or sets a Stretch enumeration that specifies how the brush's selected content (ViewBox) is displayed in the brush's tiles (ViewPort). Inherited from TileBrush.
TileMode	Gets or sets a TileMode structure that specifies how the brush's tiles fill the output area. Inherited from TileBrush.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
VerticalAlignment	Gets or sets a VerticalAlignment enumeration that specifies how the brush's content is vertically aligned within its tiles. Inherited from TileBrush.
ViewBox	Gets or sets the position and dimensions of the brush's content. Inherited from TileBrush.
ViewBoxAnimations	Gets or sets a collection of RectModifier objects that animate the ViewBox of the brush. Inherited from TileBrush.
ViewPort	Gets or sets the position and dimensions of the brush's tiles. Inherited from TileBrush.
ViewPortAnimations	Gets or sets a collection of RectModifier objects that animate the ViewPort of the brush. Inherited from TileBrush.
ViewPortUnits	Gets or sets a BrushMappingMode enumeration that specifies whether the value of the brush's ViewPort—the size and position of the brush's tiles—is relative to the size of the output area. Inherited from TileBrush.

DrawingContext Class

Definition: Drawing context.

Method	Description
Close	Closes the DrawingContext and flushes the content. Afterwards the DrawingContext can not be used anymore. This call does not require all Push calls to have been Popped.
Dispose	This is the same as the Close call: Closes the DrawingContext and flushes the content. Afterwards the DrawingContext can not be used anymore. This call does not require all Push calls to have been Popped.
DrawDrawing	Draw a Drawing at the location specified by the Point. For more fine grained control, consider filling a Rect with an RenderDataBrush via DrawRect.
DrawEllipse	Draw an ellipse with the provided Brush and/or Pen. If both the Brush and Pen are null this call is a no-op.
DrawGeometry	Draw a Geometry with the provided Brush and/or Pen. If both the Brush and Pen are null this call is a no-op.
DrawGlyphs	Draw a GlyphRun.
DrawImage	Draw an Image into the region specified by the Rect. The Image will potentially be stretched and distorted to fit the Rect. For more fine grained control, consider filling a Rect with an ImageBrush via DrawRect.
DrawLine	Draws a line with the specified pen. Note that this API does not accept a Brush, as there is no area to fill.

DrawRectangle	Draw a rectangle with the provided Brush and/or Pen. If both the Brush and Pen are null this call is a no-op.
DrawRoundedRectangle	Draw a rounded rectangle with the provided Brush and/or Pen. If both the Brush and Pen are null this call is a no-op.
DrawSubLineCollection	Draw a SubLineCollection.
DrawText	Draw Text at the location specified.
DrawVideo	Draw an Video into the region specified by the Rect. The Video will potentially be stretched and distorted to fit the Rect. For more fine grained control, consider filling a Rect with an VideoBrush via DrawRect.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
Pop	Pop the most recent Push operation, which may have been a Clip, Opacity, Transform, etc.
PushClip	Push a clip region, which will apply to all drawing primitives until the corresponding Pop call.
PushOpacity	Push an opacity which will blend the composite of all drawing primitives added until the corresponding Pop call.
PushTransform	Push a Transform which will apply to all drawing operations until the corresponding Pop.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
VerifyAPINonStructuralChange	This verifies that the API can be called for read only access.

Property	Description
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.

The following example shows the simplest way to draw text to a visual in C#. The DrawText method is used to pass FormattedText to the DrawingContext object. The DrawingContext is later drawn to the screen using the RenderOpen method of the DrawingVisual object.

```
[C#]
using System;
using MSAvalon.Windows;
using MSAvalon.Windows.Controls;
using MSAvalon.Windows.Media;
```

```

namespace DrawText_Demo
{
    public class MyApp : MS Avalon.Windows.Application
    {
        MS Avalon.Windows.Media.DrawingContext myDrawingContext;
        MS Avalon.Windows.Media.DrawingVisual myDrawingVisual;
        MS Avalon.Windows.Window mainWindow;

        protected override void OnStartingUp (StartingUpCancelEventArgs e)
        {
            base.OnStartingUp (e);
            CreateAndShowMainWindow ();
        }

        private void CreateAndShowMainWindow ()
        {
            // Create the application's main window
            mainWindow = new MS Avalon.Windows.Window ();
            mainWindow.Show ();

            // Draw the Text
            DrawingVisual myDrawingVisual = new DrawingVisual();
            DrawingContext myDrawingContext = myDrawingVisual.RenderOpen();

            myDrawingContext.DrawText(new FormattedText("Hello World!", new
Typeface("Verdana"), 14, Brushes.Black), new Point(10, 10));
            myDrawingContext.Close();

            ((IVisual)mainWindow).Children.Add(myDrawingVisual);
        }
    }

    internal sealed class EntryClass
    {
        [System.STAThread()]
        private static void Main ()
        {
            MyApp app = new MyApp ();

            app.Run ();
        }
    }
}

```

This example demonstrates how to draw using a `DrawingVisual`. There are two ways to use a `DrawingVisual`: create a class that derives from `DrawingVisual`, or create a new instance of the `DrawingVisual` class and use its `RenderOpen` method to obtain a `DrawingContext`. The latter method is demonstrated in this example.

In the following example, a `DrawingVisual` is created and used to draw an ellipse.

```

// C#
Window mainWindow = new MS Avalon.Windows.Window();
mainWindow.Show();

```

```

DrawingVisual myDrawingVisual = new DrawingVisual();
DrawingContext myDrawingContext = myDrawingVisual.RenderOpen();

SolidColorBrush mySolidColorBrush = new SolidColorBrush();
mySolidColorBrush.Color = Colors.Red;
Pen myPen = new Pen(Brushes.Blue, 10);

EllipseGeometry aGeometry = new EllipseGeometry(new Point(200,200), 100, 50);
myDrawingContext.DrawGeometry(Brushes.LimeGreen, myPen, aGeometry);
myDrawingContext.Close();

((IVisual)mainWindow).Children.Add(myDrawingVisual);

```

```

' VB .NET
mainWindow.Show()
Dim myDrawingVisual As new MS Avalon.Windows.Media.DrawingVisual
Dim myDrawingContext = myDrawingVisual.RenderOpen()

Dim mySolidColorBrush As new MS Avalon.Windows.Media.SolidColorBrush
mySolidColorBrush.Color = Colors.Red

Dim myPen As new MS Avalon.Windows.Media.Pen( _
    MS Avalon.Windows.Media.Brushes.Blue, 10)

Dim aGeometry As new MS Avalon.Windows.Media.EllipseGeometry( _
    new MS Avalon.Windows.Point(200,200), 100, 50)

myDrawingContext.DrawGeometry( _
    MS Avalon.Windows.Media.Brushes.Blue, myPen, aGeometry)
myDrawingContext.Close()

CType(mainWindow, MS Avalon.Windows.Media.IVisual).Children.Add(myDrawingVisual)

```

It is important to call the Close method of a DrawingContext after using it.

In the previous example, the application's Window is cast as a IVisual before adding the drawing visual as its child. When adding a non-element child to an element, You must cast the element as a IVisual, or else an exception will be thrown.

```

// C#
((IVisual)mainWindow).Children.Add(myDrawingVisual);

```

```

' VB .NET
CType(mainWindow, MS Avalon.Windows.Media.IVisual).Children.Add(myDrawingVisual)

```

DrawingVisual Class

Definition: Visual that contains graphical content to be drawn.

Method	Description
ClearValue	Clears the local value of a property Inherited from

DrawingVisual	DependencyObject. Initializes a new instance of the DrawingVisual class.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Releases all resources held by the Visual object. Inherited from Visual.
FindCommonVisualAncestor	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetLocalValueEnumerator	Create a local value enumerator for this instance Inherited from DependencyObject.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Retrieve the value of a property Inherited from DependencyObject.
HitTest	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
HitTestCore	HitTestCore implements precise hit testing against render contents
InvalidateProperty	Invalidates a property Inherited from DependencyObject.
IsAncestorOf	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
IsDescendantOf	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
IVisual.FindCommonVisualAncestor	Inherited from Visual.
IVisual.HitTest	Inherited from Visual.
IVisual.IsAncestorOf	Inherited from Visual.
IVisual.IsDescendantOf	Inherited from Visual.
IVisual.TransformFromAncestor	Inherited from Visual.
IVisual.TransformFromDescendant	Inherited from Visual.
IVisual.TransformFromVisual	Inherited from Visual.
IVisual.TransformToAncestor	Inherited from Visual.
IVisual.TransformToDescendant	Inherited from Visual.
IVisual.TransformToVisual	Inherited from Visual.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
OnDelayedInvalidate	TODO: Left over from WCP FastBuild, determine relevance in future version of FastBuild Inherited from DependencyObject.
OnPropertyInvalidated	Notification that a specified property has been invalidated Inherited from DependencyObject.
ReadLocalValue	Retrieve the local value of a property (if set) Inherited from DependencyObject.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
RenderOpen	Opens the DrawingVisual for rendering. The returned DrawingContext can be used to render into the DrawingVisual.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
SetValue	Sets the local value of a property Inherited from

	DependencyObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
TransformFromAncestor	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
TransformFromDescendant	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
TransformFromVisual	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
TransformToAncestor	Returns a transformation matrix that can be used to transform coordinates from this node to a specified ancestor Visual.
TransformToDescendant	Returns a transformation matrix that can be used to transform coordinates from this node to a specified descendant Visual.
TransformToVisual	Re-exposes the Visual base class's corresponding IVisual implementation as public method.
ValidateProperty	Retrieve the value of a property (for use by native cache backed custom get accessors) Inherited from DependencyObject.
ValidatePropertyCore	Allows subclasses to participate in property value computation Inherited from DependencyObject.

Property	Description
Children	Gets a collection of the DrawingVisual's children.
Clip	Gets or sets the clipping region of this DrawingVisual.
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
DependencyObjectType	Returns the DType that represents the CLR type of this instance Inherited from DependencyObject.
HasChildren	Gets a value that indicates whether the DrawingVisual has a child collection.
HitTestBounds	HitBounds returns the hit region bounding box for the current visual. Inherited from Visual.
IsDisposed	Gets a value that indicates whether the system has disposed of the Visual. Inherited from Visual.
Opacity	Gets or sets the opacity of the DrawingVisual.
Parent	Gets the parent DrawingVisual.

This example demonstrates how to draw using a DrawingVisual. There are two ways to use a DrawingVisual: create a class that derives from DrawingVisual, or create a new instance of the DrawingVisual class and use its RenderOpen method to obtain a DrawingContext. The latter method is demonstrated in this example.

In the following example, a DrawingVisual is created and used to draw an ellipse.

```
// C#
Window mainWindow = new MS Avalon.Windows.Window();
mainWindow.Show();

DrawingVisual myDrawingVisual = new DrawingVisual();
DrawingContext myDrawingContext = myDrawingVisual.RenderOpen();

SolidColorBrush mySolidColorBrush = new SolidColorBrush();
```



```

mySolidColorBrush.Color = Colors.Red;
Pen myPen = new Pen(Brushes.Blue, 10);

EllipseGeometry aGeometry = new EllipseGeometry(new Point(200,200), 100, 50);
myDrawingContext.DrawGeometry(Brushes.LimeGreen, myPen, aGeometry);
myDrawingContext.Close();

((IVisual)mainWindow).Children.Add(myDrawingVisual);

```

```

' VB .NET
mainWindow.Show()
Dim myDrawingVisual As new MS Avalon.Windows.Media.DrawingVisual
Dim myDrawingContext = myDrawingVisual.RenderOpen()

Dim mySolidColorBrush As new MS Avalon.Windows.Media.SolidColorBrush
mySolidColorBrush.Color = Colors.Red

Dim myPen As new MS Avalon.Windows.Media.Pen( _
    MS Avalon.Windows.Media.Brushes.Blue, 10)

Dim aGeometry As new MS Avalon.Windows.Media.EllipseGeometry( _
    new MS Avalon.Windows.Point(200,200), 100, 50)

myDrawingContext.DrawGeometry( _
    MS Avalon.Windows.Media.Brushes.Blue, myPen, aGeometry)
myDrawingContext.Close()

CType(mainWindow, MS Avalon.Windows.Media.IVisual).Children.Add(myDrawingVisual)

```

It is important to call the Close method of a DrawingContext after using it. In the previous example, the application's Window is cast as a IVisual before adding the drawing visual as its child. When adding a non-element child to an element, You must cast the element as a IVisual, or else an exception will be thrown.

```

// C#
((IVisual)mainWindow).Children.Add(myDrawingVisual);

```

```

' VB .NET
CType(mainWindow, MS Avalon.Windows.Media.IVisual).Children.Add(myDrawingVisual)

```

This example demonstrates how to use a PrintContext object to write to a printer. The following code demonstrates how to print a Visual (the VectorPage in this example) to the default printer. The VectorPage in this example derives from DrawingVisual and is used to draw several shapes. Two VectorPage objects are printed by calling the AddPage method twice.

[C#]

```

public static void PrintPaginatedVisualToDefaultPrinter() {
    // create a print context for my default printer
    MS Avalon.Windows.Media.PrintContext pc =
        new MS Avalon.Windows.Media.PrintContext();

```

```

// Add two pages to the default rendition
pc.AddPage(new VectorPage());
pc.AddPage(new VectorPage());

//Print
pc.Print();
}

```

The next example demonstrates how to print a Visual to a printer that the user selects. The `SelectPrintQueue` method is used to produce a dialog box that enables the user to select the printer, number of copies to print, and other options.

[C#]

```

public static void PrintPaginatedVisualToSelectedPrinter() {
// Let user select a printer queue using common dialog box
System.Printing.PrintSubSystem.PrintQueue queue =
    MSAvalon.Windows.Media.PrintContext.SelectPrintQueue();

if (queue != null) {
    // Create MSAvalon.Windows.Media.PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "Printing Example");

    // Add a page of visual
    pc.AddPage(new VectorPage());

    // Print
    pc.Print();
}
}

```

The next example demonstrates how to print to landscape and portrait page orientations. In the following code, a `JobTicket` is obtained and used to set the page orientation. The `DimensionPage` in this example is a `DrawingVisual` that adjust the size of its drawing to the size of the output page. The `GetPaperWidthHeight` method in this example is used to retrieve the page size information from the `JobTicket`.

[C#]

```

public static void MixedOrientationPrint() {
System.Printing.PrintSubSystem.LocalPrintServer ps =
    new System.Printing.PrintSubSystem.LocalPrintServer();

System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

MSAvalon.Windows.Media.PrintContext pc =
    new MSAvalon.Windows.Media.PrintContext(queue, "Landscape Example");

Microsoft.Printing.JobTicket.JobTicket jt = pc.JobTicket;

// Print a page with landscape orientation.
{
    jt.PageOrientation.Value =
        System.Printing.Configuration.PrintSchema.OrientationValues.Landscape;
}
}

```

```

        double width, height;

        GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

        // Add a page of visual
        pc.AddPage(new DimensionPage(height, width));
    }

    // Print a page with portrait orientation.
    {
        jt.PageOrientation.Value =
            System.Printing.Configuration.PrintSchema.OrientationValues.Portrait;

        double width, height;

        GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

        // Add a page of visual
        pc.AddPage(new DimensionPage(width, height));
    }

    // Print
    pc.Print();
}

internal static void GetPaperWidthHeight(Microsoft.Printing.JobTicket.JobTicket jt,
    System.Printing.PrintSubSystem.PrintQueue pq, out double width, out double height) {

    Microsoft.Printing.DeviceCapabilities.DeviceCapabilities devcap =
        new Microsoft.Printing.DeviceCapabilities.DeviceCapabilities(
            pq.AcquireDeviceCapabilities(null));

    devcap.LengthUnitType = System.Printing.Configuration.PrintSchema.LengthUnitTypes.Inch;

    width = (double) devcap.PageCanvasSizeCap.CanvasSizeX * 96;
    height = (double) devcap.PageCanvasSizeCap.CanvasSizeY * 96;
}

```

The next example demonstrates how to print based on the GetPage event. The GeneratePage method is used to handle the event.

[C#]

```

public static void EventDriven() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    // Create MSAvalon.Windows.Media.PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "EventDriven(2 pages)");

    pc.GetPage += new MSAvalon.Windows.Media.PrintContext.GetPageEventHandler(GetAPage);
}

```

```

    // Print
    pc.Print();
}

private static MSAvalon.Windows.Media.Visual GeneratePage(object sender,
int pageNo, MSAvalon.Windows.Media.GetPageEventArgs ev) {

    if (pageNo == 0) {
        return new VectorPage();
    }
    else if (pageNo == 1) {
        return new ImagePage();
    }
    else return null;
}

```

When running the sample and performing this operation, a dialog box with an assertion error message may appear. Click the Ignore All button to ignore the assertion and print the pages. The final example shows how to print to a file by using the PrintContext object's Output property.

[C#]

```

public static void PrintToFile() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    // Create PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "PrintToFile");

    pc.Output = "test.prn";

    // Add a page of visual
    pc.AddPage(new VectorPage());

    // Print
    pc.Print();
}

```

EllipseGeometry Class

Definition: Represents the geometry of a circle or ellipse.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this EllipseGeometry.
Copy	Returns a modifiable copy of the current object. The copy's

	IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this Geometry. Inherited from Geometry.
DisableCore	Inherited from Geometry.
Dispose	Releases the resources associated with the object. Inherited from Geometry.
DoesContain	Returns if point is inside the geometry. Inherited from Geometry.
EllipseGeometry	Initializes a new instance of the EllipseGeometry class.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Geometry.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Geometry.
GetBounds	Returns the bounds of a Geometry, widened according to the characteristics of the specified pen. Inherited from Geometry.
GetCurrentValue	Returns a non-animated version of this EllipseGeometry that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this Geometry that represents its current state. Inherited from Geometry.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Inherited from Geometry.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.

SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from Geometry.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
Bounds	Gets a Rect that represents the bounding box of a EllipseGeometry. This method does not take any pens into account.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Center	Gets or sets the center point of the ellipse.
CenterAnimations	Gets or sets a collection of PointModifier objects that animate the center position of the ellipse.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the geometry has animations.
IsAnimating	Returns true if the geometry contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
RadiusX	Gets or sets the x-radius value of the ellipse.
RadiusXAnimations	Gets or sets a collection of DoubleModifier objects that animate the x-radius of the ellipse.
RadiusY	Gets or sets the y-radius value of the ellipse.
RadiusYAnimations	Gets or sets a collection of DoubleModifier objects that animate the y-radius of the ellipse.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a

	DrawingContext command. Inherited from Changeable.
Transform	Gets or sets the Transform object applied to a Geometry. Inherited from Geometry.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
```

```
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();
```

```
// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));
```

```
' VB .NET
```

```
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

```
' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))
```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
```

```
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));
```

```
myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));
```

```
myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));
```

```
MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);
```

```
myPathFigure.AddSegment(myBezierSegment);
```

```
myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))
```

```
Dim myBezierSegment As new BezierSegment( _
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(200, 50), true)
```

```
myPathFigure.AddSegment(myBezierSegment)
```

```
' Add the PathFigure to the PathGeometry
myPathGeometry.Figures.Add(myPathFigure)
```


In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

' VB .NET
Dim myGeometryCollection As new MSAvalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

Dim myPath As new Path()
myPath.Data = myGeometryCollection

' Set the outline and the fill of the Path element.
myPath.Stroke = MSAvalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MSAvalon.Windows.Length(5)
Dim solidFill As new MSAvalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MSAvalon.Windows.Media.Colors.Orange, _
    MSAvalon.Windows.Media.Colors.Red)

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)
```

Geometry objects may also be rendered using the DrawingContext, which supplies a DrawGeometry method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

This example demonstrates how to draw shapes using the Geometry and Path elements in "Longhorn" markup language (code-named "XAML"). In the following example, a Path is drawn on a Canvas. Several Geometry elements are assigned to the Path element's Data attribute.

```
<Canvas ID="root"
    Background="White"
    xmlns="http://schemas.microsoft.com/2003/xaml">

    <Path ID="myPath"
```

```

Fill="Blue"
Stroke="Black"
StrokeThickness="5">

```

```

<Path.Data>
  <GeometryCollection>

    <LineGeometry StartPoint="50,50" EndPoint="300,50"/>
    <EllipseGeometry Center="440, 100" RadiusX="40" RadiusY="75"/>
    <RectangleGeometry >
      <RectangleGeometry.Rect>
        <Rect X="400" Y="225" Width="100" Height="50"/>
      </RectangleGeometry.Rect>
    </RectangleGeometry>

    <PathGeometry>
      <PathGeometry.Figures>
        <PathFigureCollection>
          <PathFigure>
            <PathFigure.Segments>
              <PathSegmentCollection>
                <StartSegment Point="400,100"/>
                <BezierSegment Point1="400,100" Point2="400,200" Point3="200,300"/>
                <BezierSegment Point1="400,300" Point2="400,100" Point3="200,50"/>
                <BezierSegment Point1="0,100" Point2="0,200" Point3="200,300"/>
                <BezierSegment Point1="0,300" Point2="0,100" Point3="200,50"/>
              </PathSegmentCollection>
            </PathFigure.Segments>
          </PathFigure>
        </PathFigureCollection>
      </PathGeometry.Figures>
    </PathGeometry>

  </GeometryCollection>

</Path.Data>
</Path>

```

```

</Canvas>

```

In the previous example, the PathFigure object, one of the shapes drawn inside the Path element, contains a StartSegment but no CloseSegment; if a CloseSegment were added to the figure, a line would be drawn from the last segment in the collection back to the starting point of the figure.

This example demonstrates how to animate a Geometry. In the following example, a PointAnimation is used to animate the Center of an EllipseGeometry.

```

<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <!-- Use a Path to draw the Geometry. -->

  <Path ID="myPath"
    Fill="Blue"
    Stroke="Black"

```

```

StrokeThickness="5">
<Path.Data>
<!-- Draw an Ellipse. -->
<EllipseGeometry Center="100,100" RadiusX="25" RadiusY="100">

    <EllipseGeometry.CenterAnimations>
    <!--Animate the center of this ellipse: -->
    <PointAnimation From="100,100" To="50,50"
        Duration="5" Begin="0" AutoReverse="true" RepeatCount="20"/>
    </EllipseGeometry.CenterAnimations>

</EllipseGeometry>
</Path.Data>
</Path>
</Canvas>

```

In the previous example, the `PointAnimationCollection` tag, `<PointAnimationCollection>`, is omitted when animating the ellipse's center. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `CenterAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag. For more information about animating properties, see [Animation in "Avalon"](#).

```

// C#
EllipseGeometry myEllipseGeometry = new EllipseGeometry();
myEllipseGeometry.Center = new Point(200,200);
myEllipseGeometry.RadiusX = 25;
myEllipseGeometry.RadiusY = 50;

PointAnimation myPointAnimation = new PointAnimation();
myPointAnimation.From = new Point(200,200);
myPointAnimation.To = new Point(50,50);
myPointAnimation.Duration = new Time(5000);
myPointAnimation.Begin = new TimeSyncValue(new Time(0));
myPointAnimation.AutoReverse = true;
myPointAnimation.RepeatCount = 20;

myEllipseGeometry.CenterAnimations.Add(myPointAnimation);

```

```

' VB .NET
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry
myEllipseGeometry.Center = new MSAvalon.Windows.Point(200,200)
myEllipseGeometry.RadiusX = 25
myEllipseGeometry.RadiusY = 50

Dim myPointAnimation As new MSAvalon.Windows.Media.Animation.PointAnimation
myPointAnimation.From = new MSAvalon.Windows.Point(200,200)
myPointAnimation.To = new MSAvalon.Windows.Point(50,50)
myPointAnimation.Duration = new Time(5000)
myPointAnimation.Begin = new TimeSyncValue(new Time(0))
myPointAnimation.AutoReverse = true
myPointAnimation.RepeatCount = 20

myEllipseGeometry.CenterAnimations.Add(myPointAnimation)

```

F ntFamily Class

Definition: Font Family.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
FontFamily	Construct a font family
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Baseline	Distance from character cell top to English baseline relative to em size.
FamilyName	Font family name
Height	Full height of character cell relative to em size

FormattedText Class

Definition: The FormattedText class is a part of Avalon MIL easy text API, which is targeted at programmers needing to add some simple text to a MIL visual.

Method	Description
BuildGeometry	Obtains geometry for the text, including underlines and strikethroughs.
BuildHighlightGeometry	Builds a highlight geometry object.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
FormattedText	Construct a FormattedText object
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetMaxTextWidths	Obtains a copy of the array of lengths, which will be applied to each line of text in turn. If the text covers more lines than there are entries in the length array, the last entry is reused as many times as required. The maxTextWidths array overrides the MaxTextWidth property.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetFontFamily	Sets or changes the font family for the text object
SetFontSize	Sets or changes the font em size measured in MIL units

SetFontStretch	Sets or changes the font stretch
SetFontStyle	Sets or changes the font style
SetFontTypeface	Sets or changes the type face
SetFontWeight	Sets or changes the font weight
SetForegroundBrush	Sets foreground brush used for drawing text
SetMaxTextWidths	Sets the array of lengths, which will be applied to each line of text in turn. If the text covers more lines than there are entries in the length array, the last entry is reused as many times as required. The maxTextWidths array overrides the MaxTextWidth property.
SetTextDecorations	Sets or changes the text decorations
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Baseline	The distance from the top of the first line to the baseline of the first line.
Extent	The distance from the topmost black pixel of the first line to the bottommost black pixel of the last line.
Height	The distance from the top of the first line to the bottom of the last line.
LineHeight	Defines the height of each line, and thus the spacing between lines. Default value: Line height defaults to the recommended line spacing of the font passed to the FormattedText constructor.
MaxLineCount	Defines the maximum number of lines to display. The last line of text displayed is the lineCount-1'th line, or the last whole line that will fit within the count set by MaxTextHeight, whichever occurs first. Use the Trimming property to control how the omission of text is indicated
MaxTextHeight	Sets the maximum length of a column of text. The last line of text displayed is the last whole line that will fit within this limit, or the nth line as specified by MaxLineCount, whichever occurs first. Use the Trimming property to control how the omission of text is indicated.
MaxTextWidth	Defines the maximum length of lines. Text will be word-wrapped if necessary to avoid exceeding this limit. Note that it is glyph alignment edges that are kept within the line count limit. Many fonts contain glyphs that overhang a little beyond their alignment edges. Clients should allow at least 1/6 em (font size) margin beyond the line count requested to leave room for such overhangs. A zero value for line count is treated as an infinite line count.
MinWidth	The minimum line count that can be specified without causing any word to break.
OverhangAfter	The distance from the bottom of the last line to the extent bottom.
OverhangLeading	The maximum distance from the leading black pixel to the leading alignment point of a line.
OverhangTrailing	The maximum distance from the trailing black pixel to the trailing alignment point of a line.
Text	Returns the string of text to be displayed
Trimming	Defines how omission of text is indicated. CharacterEllipsis trimming allows partial words to be displayed, while WordEllipsis removes whole words to fit. Both guarantee to include an ellipsis ('...') at the end of the lines where text has been trimmed as a result of line and column limits.
Width	The maximum advance width between the leading and trailing alignment points of a line, excluding the width of whitespace characters at the end of the line.

The maximum advance width between the leading and trailing alignment points of a line, including the width of whitespace characters at the end of the line.

The following example shows the simplest way to draw text to a visual in C#. The DrawText method is used to pass FormattedText to the DrawingContext object. The DrawingContext is later drawn to the screen using the RenderOpen method of the DrawingVisual object.

```
[C#]
using System;
using MS Avalon.Windows;
using MS Avalon.Windows.Controls;
using MS Avalon.Windows.Media;

namespace DrawText_Demo
{
    public class MyApp : MS Avalon.Windows.Application
    {
        MS Avalon.Windows.Media.DrawingContext myDrawingContext;
        MS Avalon.Windows.Media.DrawingVisual myDrawingVisual;
        MS Avalon.Windows.Window mainWindow;

        protected override void OnStartingUp (StartingUpCancelEventArgs e)
        {
            base.OnStartingUp (e);
            CreateAndShowMainWindow ();
        }

        private void CreateAndShowMainWindow ()
        {
            // Create the application's main window
            mainWindow = new MS Avalon.Windows.Window ();
            mainWindow.Show ();

            // Draw the Text
            DrawingVisual myDrawingVisual = new DrawingVisual();
            DrawingContext myDrawingContext = myDrawingVisual.RenderOpen();

            myDrawingContext.DrawText(new FormattedText("Hello World!", new
Typeface("Verdana"), 14, Brushes.Black), new Point(10, 10));
            myDrawingContext.Close();

            ((IVisual)mainWindow).Children.Add(myDrawingVisual);
        }
    }

    internal sealed class EntryClass
    {
        [System.STAThread()]
        private static void Main ()
        {
            MyApp app = new MyApp ();

            app.Run ();
        }
    }
}
```

}

Geometry Class

Definition: An abstract class that provides base functionality for all geometry classes, such as EllipseGeometry, RectangleGeometry, and PathGeometry. The Geometry class of objects can be used for clipping, hit-testing, and rendering 2-D graphic data.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Subclasses must implement this to provide clones of themselves. Inherited from Animatable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this Geometry.
DisableCore	
Dispose	Releases the resources associated with the object.
DoesContain	Returns if point is inside the geometry.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	
GetBounds	Returns the bounds of a Geometry, widened according to the characteristics of the specified pen.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this Geometry that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable.

	<p>If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code>.</p> <p>Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code>.</p>
<code>OnChanged</code>	
<code>PropagateEventHandler</code>	
<code>ReadPreamble</code>	<p>Ensures that simple (non-<code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code>.</p>
<code>ReferenceEquals</code>	<p>Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code>.</p>
<code>SetDefaultParentTimeline</code>	<p>This will change the time parent for animations on any of the properties of this <code>Changeable</code> which are inheriting their time parent. Inherited from <code>Animatable</code>.</p>
<code>SetDefaultParentTimelineCore</code>	
<code>ToString</code>	<p>Returns a <code>String</code> that represents the current <code>Object</code>. Inherited from <code>Object</code>.</p>
<code>ValidateObjectState</code>	<p>Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code>.</p>
<code>WritePostscript</code>	<p>Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code>.</p>
<code>WritePreamble</code>	<p>Ensures that simple (non-<code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from <code>Changeable</code>.</p>

Property	Description
<code>AllowChangeableReferenceOverride</code>	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
<code>Bounds</code>	Gets a <code>Rect</code> that specifies the bounding box of a <code>Geometry</code> . This method does not take any pens into account.
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>DefaultParentTimeline</code>	The current parent <code>Timeline</code> associated with this <code>Animatable</code> . This will be the <code>Timeline</code> set to the <code>ParentTimeline</code> property of this <code>Animatable</code> if one has been set and if not, the <code>Timeline</code> last passed into the <code>SetDefaultParentTimeline</code> method. Inherited from <code>Animatable</code> .
<code>HasAnimations</code>	Gets a <code>Boolean</code> that indicates whether the geometry has animations.
<code>IsAnimating</code>	Returns true if the geometry contains animations that are active.
<code>IsChangeable</code>	Gets a <code>Boolean</code> that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
<code>IsOverridingBaseValue</code>	
<code>StatusOfNextUse</code>	Gets or sets a <code>UseStatus</code> enumeration that specifies how the

	Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets the Transform object applied to a Geometry.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's

StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();

// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));
```

```
' VB .NET
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()

' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))
```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));

myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));

myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));

MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);

myPathFigure.AddSegment(myBezierSegment);

myPathGeometry.Figures.Add(myPathFigure);

' VB .NET
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))

Dim myBezierSegment As new BezierSegment( _
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(200, 50), true)

myPathFigure.AddSegment(myBezierSegment)
```

```

    Add the PathFigure to the PathGeometry
    myPathGeometry.Figures.Add(myPathFigure)

```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```

// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

' VB .NET
Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

Dim myPath As new Path()
myPath.Data = myGeometryCollection

' Set the outline and the fill of the Path element.
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _
    MS Avalon.Windows.Media.Colors.Red)

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)

```

Geometry objects may also be rendered using the DrawingContext, which supplies a DrawGeometry method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

This example demonstrates how to draw shapes using the Geometry and Path elements in "Longhorn" markup language (code-named "XAML").

In the following example, a Path is drawn on a Canvas. Several Geometry elements are assigned to the Path element's Data attribute.

```

<Canvas ID="root"
  Background="White"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Path ID="myPath"
    Fill="Blue"
    Stroke="Black"
    StrokeThickness="5">

    <Path.Data>
      <GeometryCollection>

        <LineGeometry StartPoint="50,50" EndPoint="300,50"/>
        <EllipseGeometry Center="440, 100" RadiusX="40" RadiusY="75"/>
        <RectangleGeometry >
          <RectangleGeometry.Rect>
            <Rect X="400" Y="225" Width="100" Height="50"/>
          </RectangleGeometry.Rect>
        </RectangleGeometry>

        <PathGeometry>
          <PathGeometry.Figures>
            <PathFigureCollection>
              <PathFigure>
                <PathFigure.Segments>
                  <PathSegmentCollection>
                    <StartSegment Point="400,100"/>
                    <BezierSegment Point1="400,100" Point2="400,200" Point3="200,300"/>
                    <BezierSegment Point1="400,300" Point2="400,100" Point3="200,50"/>
                    <BezierSegment Point1="0,100" Point2="0,200" Point3="200,300"/>
                    <BezierSegment Point1="0,300" Point2="0,100" Point3="200,50"/>
                  </PathSegmentCollection>
                </PathFigure.Segments>
              </PathFigure>
            </PathFigureCollection>
          </PathGeometry.Figures>
        </PathGeometry>

      </GeometryCollection>

    </Path.Data>
  </Path>

</Canvas>

```

In the previous example, the PathFigure object, one of the shapes drawn inside the Path element, contains a StartSegment but no CloseSegment; if a CloseSegment were added to the figure, a line would be drawn from the last segment in the collection back to the starting point of the figure.

This example demonstrates how to animate a Geometry. In the following example, a PointAnimation is used to animate the Center of an EllipseGeometry.

```

<Canvas ID="root"

```

```

xmlns="http://schemas.microsoft.com/2003/xaml">

<!-- Use a Path to draw the Geometry. -->

<Path ID="myPath"
      Fill="Blue"
      Stroke="Black"
      StrokeThickness="5">
  <Path.Data>
    <!-- Draw an Ellipse. -->
    <EllipseGeometry Center="100,100" RadiusX="25" RadiusY="100">

      <EllipseGeometry.CenterAnimations>
        <!--Animate the center of this ellipse: -->
        <PointAnimation From="100,100" To="50,50"
          Duration="5" Begin="0" AutoReverse="true" RepeatCount="20"/>
      </EllipseGeometry.CenterAnimations>

    </EllipseGeometry>
  </Path.Data>
</Path>
</Canvas>

```

In the previous example, the `PointAnimationCollection` tag, `<PointAnimationCollection>`, is omitted when animating the ellipse's center. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `CenterAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag. For more information about animating properties, see [Animation](#) in "Avalon".

```

// C#
EllipseGeometry myEllipseGeometry = new EllipseGeometry();
myEllipseGeometry.Center = new Point(200,200);
myEllipseGeometry.RadiusX = 25;
myEllipseGeometry.RadiusY = 50;

PointAnimation myPointAnimation = new PointAnimation();
myPointAnimation.From = new Point(200,200);
myPointAnimation.To = new Point(50,50);
myPointAnimation.Duration = new Time(5000);
myPointAnimation.Begin = new TimeSyncValue(new Time(0));
myPointAnimation.AutoReverse = true;
myPointAnimation.RepeatCount = 20;

myEllipseGeometry.CenterAnimations.Add(myPointAnimation)

' VB .NET
Dim myEllipseGeometry As new MS Avalon.Windows.Media.EllipseGeometry
myEllipseGeometry.Center = new MS Avalon.Windows.Point(200,200)
myEllipseGeometry.RadiusX = 25
myEllipseGeometry.RadiusY = 50

Dim myPointAnimation As new MS Avalon.Windows.Media.Animation.PointAnimation
myPointAnimation.From = new MS Avalon.Windows.Point(200,200)

```

```

myPointAnimation.To = new MS Avalon.Windows.Point(50,50)
myPointAnimation.Duration = new Time(5000)
myPointAnimation.Begin = new TimeSyncValue(new Time(0))
myPointAnimation.AutoReverse = true
myPointAnimation.RepeatCount = 20

```

```

myEllipseGeometry.CenterAnimations.Add(myPointAnimation)

```

GeometryCollection Class

Definition: Represents a collection of Geometry objects.

Method	Description
Add	
AddRange	
Clear	
CloneCore	Implementation of Animatable.CloneCore.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	
Copy	Creates a copy of this Geometry. Inherited from Geometry.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this GeometryCollection.
CopyTo	
DisableCore	
Dispose	Dispose resources associated with geometry.
DoesContain	Returns if point is inside the geometry. Inherited from Geometry.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Geometry.
GeometryCollection	
GetBounds	Returns the bounds of a Geometry, widened according to the characteristics of the specified pen. Inherited from Geometry.
GetCurrentValue	Returns a non-animated version of this GeometryCollection that represents its current state.
GetCurrentValue	Returns a non-animated version of this Geometry that represents its current state. Inherited from Geometry.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.

GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetOptimizedGeometry	
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
SetRange	
ToString	Returns a String that represents the current Object. Inherited

	from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
Bounds	Gets a Rect that specifies the bounding box of a Geometry. This method does not take any pens into account. Inherited from Geometry.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	
CombineMode	
Count	
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
IsOverridingBaseValue	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets the Transform object applied to a Geometry. Inherited from Geometry.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();

// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));

' VB .NET
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

' PathFigure objects must have a defined start point before
 ' other segments can be added.
 myPathFigure.StartAt(new MS Avalon.Windows.Point(200,50))

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));

myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));

myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));

MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);

myPathFigure.AddSegment(myBezierSegment);

myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))

Dim myBezierSegment As new BezierSegment( _
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(200, 50), true)

myPathFigure.AddSegment(myBezierSegment)

' Add the PathFigure to the PathGeometry
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);
```

```

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

```

' VB .NET

```

Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

```

```

Dim myPath As new Path()
myPath.Data = myGeometryCollection

```

```

' Set the outline and the fill of the Path element.
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _
    MS Avalon.Windows.Media.Colors.Red)

```

```

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)

```

Geometry objects may also be rendered using the `DrawingContext`, which supplies a `DrawGeometry` method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

GetPageEventArgs Class

Definition: class `GetPageEventArgs`

Method	Description
<code>Equals</code>	Determines whether two Object instances are equal. Inherited from Object.
<code>Finalize</code>	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
<code>GetHashCode</code>	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
<code>GetPageEventArgs</code>	Constructor
<code>GetType</code>	Gets the Type of the current instance. Inherited from Object.
<code>MemberwiseClone</code>	Creates a shallow copy of the current Object. Inherited from Object.
<code>ReferenceEquals</code>	Determines whether the specified Object instances are the same instance. Inherited from Object.

ToString Returns a String that represents the current Object. Inherited from Object.

Property	Description
JobTicket	JobTicket property

GlyphRun Class

Definition: Glyph Run Class.

Method	Description
BuildGeometry	Obtains geometry for the glyph run.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Finalizer
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GlyphRun	Constructs a glyph run from a string
IsEmpty	Returns whether GlyphRun doesn't contain any glyphs
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
AdvanceWidth	Advance width from origin of first glyph to far alignment edge of last glyph.
Ascent	Distance from the GlyphRun origin to the top of the alignment box.
BackgroundBrush	Background brush, can be null
BidiLevel	Determines LTR/RTL reading order and bidi nesting.
CharacterCount	The number of characters in the glyph run
ClusterMapOffset	Index of initial element in CharacterToGlyphMap
FontFace	Physical font file specification
FontHintingEmSize	Em size used for hinting
FontRenderingEmSize	Em size used for rendering
ForegroundBrush	Foreground brush
GlyphCount	The number of glyphs in the glyph run
Height	Distance from top to bottom of alignment box.
InkBoundingBox	Computes ink bounding box for the glyph run. The rectangle is relative to the glyph run origin.
Origin	Glyph run origin
StyleSimulations	Style simulation flags

GlyphTypeface Class

Definition: Physical font face corresponds to a font file on the disk.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup

	operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetGlyphOutline	Returns a geometry describing the path for a single glyph in the font. The path represents the glyph without grid fitting applied for rendering at a specific resolution.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GlyphTypeface	Constructs a GlyphTypeface from a Uri
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
AdvanceHeights	Returns Advance height for a given glyph (Used for example in vertical layout).
AdvanceWidths	Returns advance width for a given glyph.
Baseline	Distance from cell top to English baseline relative to em size.
BottomSideBearings	Distance from bottom edge of black box to bottom end of advance vector. Positive when bottom edge of black box is within the alignment rectangle defined by the advance width and font cell height. (The font cell height is a horizontal dimension in vertical layout). Negative when bottom edge of black box overhangs the alignment rectangle.
CapsHeight	Distance from baseline to top of English capital, relative to em size.
CharacterToGlyphMap	Returns nominal mapping of Unicode codepoint to glyph index as defined by the font 'CMAP' table.
Copyrights	This property is indexed by a Culture Identifier. Copyright notice.
Descriptions	This property is indexed by a Culture Identifier. Description of the typeface. Can contain revision information, usage recommendations, history, features, etc.
DesignerNames	This property is indexed by a Culture Identifier. Name of the designer of the typeface.
DesignerUrls	This property is indexed by a Culture Identifier. URL of typeface designer (with protocol, e.g., http://, ftp://).
DistancesFromBlackBoxLeftToVerticalBaseline	Offset across from left edge of black box to East Asian vertical baseline.
DistancesFromHorizontalBaselineToBlackBoxBottom	Offset down from horizontal Western baseline to bottom of glyph black box.
FaceNames	This property is indexed by a Culture Identifier. It returns the face name in the specified language, or, if the font does not provide a name for the specified language, it returns the face name in English. The

FamilyNames	face name may identify weight, style and/or stretch. This property is indexed by a Culture Identifier. It returns the family name in the specified language, or, if the font does not provide a name for the specified language, it returns the family name in English. The family name excludes weight, style and stretch.
Height	Height of character cell relative to em size.
LeftSideBearings	Distance from leading end of advance vector to left edge of black box. Positive when left edge of black box is within the alignment rectangle defined by the advance width and font cell height. Negative when left edge of black box overhangs the alignment rectangle.
LicenseDescriptions	This property is indexed by a Culture Identifier. Description of how the font may be legally used, or different example scenarios for licensed use. This field should be written in plain language, not legalese.
ManufacturerNames	This property is indexed by a Culture Identifier. Manufacturer Name.
RightSideBearings	Distance from right edge of black box to right end of advance vector. Positive when trailing edge of black box is within the alignment rectangle defined by the advance width and font cell height. Negative when right edge of black box overhangs the alignment rectangle.
SampleTexts	This property is indexed by a Culture Identifier. This can be the font name, or any other text that the designer thinks is the best sample to display the font in.
StrikeoutPosition	Position of strikeout relative to baseline relative to em size. The value is usually positive, to place the strikeout above the baseline.
StrikeoutThickness	Thickness of strikeout relative to em size.
Symbol	Returns true if this font does not conform to Unicode encoding: it may be considered as a simple collection of symbols indexed by a codepoint.
TopSideBearings	Distance from top end of (vertical) advance vector to top edge of black box. Positive when top edge of black box is within the alignment rectangle defined by the advance height and font cell height. (The font cell height is a horizontal dimension in vertical layout). Negative when top edge of black box overhangs the alignment rectangle.
Trademarks	This property is indexed by a Culture Identifier. This is used to save any trademark notice/information for this font. Such information should be based on legal advice. This is distinctly separate from the copyright.
UnderlinePosition	Position of underline relative to baseline relative to

UnderlineThickness	em size. The value is usually negative, to place the underline below the baseline. Thickness of underline relative to em size.
VendorUrls	This property is indexed by a Culture Identifier. URL of font vendor (with protocol, e.g., http://, ftp://). If a unique serial number is embedded in the URL, it can be used to register the font.
Version	Font face version interpreted from the font's 'NAME' table. This property is indexed by a Culture Identifier.
VersionStrings	Version string in the fonts NAME table. Version strings vary significantly in format - to obtain the version as a numeric value use the 'Version' property, do not attempt to parse the VersionString.
Win32FaceNames	This property is indexed by a Culture Identifier. It returns the face name in the specified language, or, if the font does not provide a name for the specified language, it returns the face name in English. The Win32Face name may identify weights other than regular or bold and/or style, but may not identify stretch or other weights.
Win32FamilyNames	This property is indexed by a Culture Identifier. It returns the family name in the specified language, or, if the font does not provide a name for the specified language, it returns the family name in English. The Win32FamilyName name excludes regular or bold weights and style, but includes other weights and stretch.
XHeight	Western x-height relative to em size.

GradientBrush Class

Definition: An abstract class that describes a gradient fill. Classes that derive from GradientBrush describe different ways of interpreting gradient stops.

Method	Description
AddStop	Adds a gradient stop to the brush.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Subclasses must implement this to provide clones of themselves. Inherited from Animatable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable.
Copy	Returns a modifiable copy of the current object. The copy's

	IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this GradientBrush that represents its current state.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
GradientBrush	Protected constructor for GradientBrush. Sets all values to their defaults. To set property values, use the constructor which accepts paramters
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called

	before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
ColorInterpolationMode	ColorInterpolationMode - Read only accessor of the ColorInterpolationMode property. This property controls how the colors in Gradient are interpolated. Default is ColorInterpolationMode.PerceptuallyLinearGamma.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
GradientStops	Gets or sets the gradient stops (transition points) of a brush.
HasAnimations	Gets a Boolean that indicates whether the brush has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
MappingMode	Gets or sets a BrushMappingMode enumeration that specifies whether the gradient brush's positioning coordinates are absolute or relative to the output area.
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.

SpreadMethod	Gets or sets the type of spread method that specifies how to draw a gradient that starts or ends inside the bounds of the object to be painted.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to use the Transform property of brushes to apply transformations to LinearGradientBrush and RadialGradientBrush fills. A LinearGradientBrush is used to fill the first two Rectangle elements. The difference between the rectangles is that the LinearGradientBrush in the second rectangle is rotated 45 degrees. The second pair of rectangles illustrates the before and after effect of a ScaleTransform by reducing a RadialGradientBrush to half its normal height.

```

    <Border xmlns="http://schemas.microsoft.com/2003/xaml"
    Background="#CCCCCC">

    <Canvas Height="40">

    <!-- Rectangle #1 is filled with a LinearGradientBrush. The gradient colors
    flow from left to right by default. -->

    <Rectangle RectangleLeft="10" RectangleTop="10"
    RectangleWidth="300" RectangleHeight="200">

    <Rectangle.Fill>
    <LinearGradientBrush>

    <LinearGradientBrush.GradientStops>
    <GradientStopCollection>
    <GradientStop Color="red" Offset="0"/>
    <GradientStop Color="yellow" Offset="1" />
    <GradientStop Color="blue" Offset="0.5"/>
    <GradientStop Color="white" Offset="0.2"/>
    </GradientStopCollection>
    </LinearGradientBrush.GradientStops>

    </LinearGradientBrush>
    </Rectangle.Fill>

    </Rectangle>

    <!-- Rectangle #2 is identical to the first rectangle except that the Transform
    property rotates the LinearGradientBrush so that the gradient colors are
    rotated by 45 degrees. -->

    <Rectangle RectangleLeft="320" RectangleTop="10"

```

```

RectangleWidth="300" RectangleHeight="200">

<Rectangle.Fill>
  <LinearGradientBrush>

    <LinearGradientBrush.Transform>
      <RotateTransform Angle="45" /> <!-- Rotation angle. -->
    </LinearGradientBrush.Transform>

    <LinearGradientBrush.GradientStops>
      <GradientStopCollection>
        <GradientStop Color="red" Offset="0"/>
        <GradientStop Color="yellow" Offset="1" />
        <GradientStop Color="blue" Offset="0.5"/>
        <GradientStop Color="white" Offset="0.2"/>
      </GradientStopCollection>
    </LinearGradientBrush.GradientStops>

  </LinearGradientBrush>
</Rectangle.Fill>

</Rectangle>

<!-- Rectangle #3 is filled with a RadialGradientBrush. -->

<Rectangle RectangleLeft="10" RectangleTop="250"
  RectangleWidth="300" RectangleHeight="200">

  <Rectangle.Fill>
    <RadialGradientBrush Focus="0.5,0.5">

      <RadialGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="red" Offset="0"/>
          <GradientStop Color="yellow" Offset="1"/>
          <GradientStop Color="blue" Offset="0.5"/>
        </GradientStopCollection>
      </RadialGradientBrush.GradientStops>

    </RadialGradientBrush>
  </Rectangle.Fill>

</Rectangle>

<!-- Rectangle #4 is identical to the third rectangle except that the Transform
property applies a ScaleTransform to the RadialGradientBrush so that the
gradient is half its previous height. -->

<Rectangle RectangleLeft="320" RectangleTop="250"
  RectangleWidth="300" RectangleHeight="200">

  <Rectangle.Fill>

    <RadialGradientBrush Focus="0.5,0.5">

```

```

<RadialGradientBrush.Transform>
  <ScaleTransform ScaleX="1" ScaleY="0.5" /><!-- Scale transform. -->
</RadialGradientBrush.Transform>

<RadialGradientBrush.GradientStops>
  <GradientStopCollection>
    <GradientStop Color="red" Offset="0"/>
    <GradientStop Color="yellow" Offset="1"/>
    <GradientStop Color="blue" Offset="0.5"/>
  </GradientStopCollection>
</RadialGradientBrush.GradientStops>

</RadialGradientBrush>
</Rectangle.Fill>

</Rectangle>

</Canvas>
</Border>

```

This example creates simple vertical, horizontal, and radial gradients and uses them to fill an element using "Longhorn" markup language (code-named "XAML").

In the following example, vertical and horizontal gradients are used to set the Fill property of two Rectangle elements. In this particular example, the gradients are described using simple notation: GradientType StartColor EndColor, where GradientType is VerticalGradient, HorizontalGradient, or RadialGradient. StartColor and EndColor can be predefined color names (such as Blue) or hexadecimal values.

```

<Canvas xmlns="http://schemas.microsoft.com/2003/xaml">

  <Rectangle
    Fill="VerticalGradient Blue Green"
    RectangleLeft="20"
    RectangleTop="20"
    RectangleWidth="100"
    RectangleHeight="100">
  </Rectangle>

  <Rectangle
    Fill="HorizontalGradient Blue Red"
    RectangleLeft="120"
    RectangleTop="120"
    RectangleWidth="100"
    RectangleHeight="100">
  </Rectangle>

```

A vertical gradient is a linear gradient whose start and endpoints form a vertical line; likewise, a horizontal gradient is a linear gradient whose start and endpoints form a horizontal line. You can explicitly describe your own linear gradients using the following syntax: LinearGradient StartPoint EndPoint StartColor EndColor, where StartPoint and EndPoint are the starting and ending coordinates, with each coordinate expressed as a pair of x and y values from 0 to 1, such as 0.1,0.1 and 0.5,0.5. These values indicate the relative position of the start or end point. An endpoint of 0.5,0.5 would be located 50 percent to the right of the fill area and 50 percent of the way from the top of the area—the middle of the shape.

In the following example, the Fill property of a Rectangle element is set by explicitly using a linear gradient.

```
<Rectangle
  Fill="LinearGradient 0.1,0.1 0.5,0.5 Blue Green"
  RectangleLeft="220"
  RectangleTop="220"
  RectangleWidth="100"
  RectangleHeight="100">
</Rectangle>
```

In the final example, the Fill property of a Rectangle element is set using a radial gradient.

```
<Rectangle
  Fill="RadialGradient Blue Red"
  RectangleLeft="320"
  RectangleTop="320"
  RectangleWidth="100"
  RectangleHeight="100">
</Rectangle>
```

</Canvas>

See [Create a Gradient with More Than Two Colors](#) for an example of how to create a gradient with more than two gradient stops.

To create vertical and horizontal gradients in code or using compound notation, use the `LinearGradientBrush` class and set its `StartPoint` and `EndPoint` properties so that they describe a vertical or horizontal line. To create radial gradients in code or using compound notation, use the `RadialGradientBrush` class.

This example demonstrates how to create a gradient that has more than two colors in "XAML". To create a gradient with more than two colors, add a `GradientStopCollection` to the gradient's `GradientStops` property. Next, add `GradientStop` objects to the `GradientStopCollection`, one for each color the gradient should contain. Set the `Color` and the `Offset`, a value from 0 to 1 that determines the relative position of the stop in the gradient, of each of the stops. The following example shows a `Button` whose `Background` is filled with a horizontal gradient that has four colors.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Button
    Canvas.Top="50"
    Canvas.Left="50"
    BorderBrush="Black"
    Width="200"
    Height="30">
    <Button.Background>
      <LinearGradientBrush >
        <LinearGradientBrush.GradientStops>
          <GradientStopCollection>
            <GradientStop Color="Red" Offset="0" />
            <GradientStop Color="Blue" Offset="0.25"/>
            <GradientStop Color="Orange" Offset="0.75"/>
            <GradientStop Color="Yellow" Offset="1"/>
          </GradientStopCollection>
        </LinearGradientBrush.GradientStops>
      </LinearGradientBrush>
    </Button.Background>
  </Button>
</Canvas>
```

```

    </Button.Background>
  </Button>

</Canvas>

```

GradientStop Class

Definition: Describes the location and color of a transition point in a gradient.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this GradientStop.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GradientStop	Initializes a new instance of the GradientStop class.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from

	Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Color	Gets or sets the color value of the gradient stop.
ColorAnimations	Gets or sets the animations associated with the Color of the gradient stop.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the gradient stop has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
Offset	Gets the location of the gradient stop within the gradient vector.

OffsetAnimations	Gets or sets a collection of objects that animate the gradient stop's offset.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to create a gradient that has more than two colors in "Longhorn" markup language (code-named "XAML"). To create a gradient with more than two colors, add a GradientStopCollection to the gradient's GradientStops property. Next, add GradientStop objects to the GradientStopCollection, one for each color the gradient should contain. Set the Color and the Offset, a value from 0 to 1 that determines the relative position of the stop in the gradient, of each of the stops. The following example shows a Button whose Background is filled with a horizontal gradient that has four colors.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button
Canvas.Top="50"
Canvas.Left="50"
BorderBrush="Black"
Width="200"
Height="30">
<Button.Background>
<LinearGradientBrush >
<LinearGradientBrush.GradientStops>
<GradientStopCollection>
<GradientStop Color="Red" Offset="0" />
<GradientStop Color="Blue" Offset="0.25"/>
<GradientStop Color="Orange" Offset="0.75"/>
<GradientStop Color="Yellow" Offset="1"/>
</GradientStopCollection>
</LinearGradientBrush.GradientStops>
</LinearGradientBrush>
</Button.Background>
</Button>

</Canvas>
```

GradientStopCollection Class

Definition: Represents a collection of GradientStop gradient stops.

Method	Description
Add	Adds a GradientStop to the gradient stop collection.
AddRange	
Clear	Removes all items from the gradient stop list.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that

CloneCore	derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Determines whether the collection contains the specified GradientStop.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	
CopyTo	Copies the entire GradientStopCollection to a compatible one-dimensional Array, starting at the specified index of the target array.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStopCollection that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetEnumerator	Returns an enumerator that can iterate through the entire GradientStopCollection collection.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
GradientStopCollection	Initializes a new instance of the GradientStopCollection class.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	Adds an item to the gradient stop list. Functions identically to Add, except that an exception is returned if the item is not the proper type.
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	Returns the zero-based index of the specified GradientStop.
Insert	Inserts a GradientStop at the specified position in the gradient stop list.

InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	Removes the first occurrence of the specified GradientStop from the collection.
RemoveAt	Removes the item at the specified index in the collection.
RemoveRange	
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from

Capacity	Changeable.
Count	Gets the number of items contained in the gradient stop collection.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	Gets a value that indicates whether the collection has a fixed size.
IsOverridingBaseValue	
Item	Gets or sets the GradientStop at the specified index in the collection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to create a gradient that has more than two colors in "Longhorn" markup language (code-named "XAML"). To create a gradient with more than two colors, add a GradientStopCollection to the gradient's GradientStops property. Next, add GradientStop objects to the GradientStopCollection, one for each color the gradient should contain. Set the Color and the Offset, a value from 0 to 1 that determines the relative position of the stop in the gradient, of each of the stops. The following example shows a Button whose Background is filled with a horizontal gradient that has four colors.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button
  Canvas.Top="50"
  Canvas.Left="50"
  BorderBrush="Black"
  Width="200"
  Height="30">
  <Button.Background>
    <LinearGradientBrush >
      <LinearGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="Red" Offset="0" />
          <GradientStop Color="Blue" Offset="0.25"/>
          <GradientStop Color="Orange" Offset="0.75"/>
          <GradientStop Color="Yellow" Offset="1"/>
        </GradientStopCollection>
      </LinearGradientBrush.GradientStops>
    </LinearGradientBrush>
  </Button.Background>
</Button>
</Canvas>
```

```

</Button.Background>
</Button>

```

```

</Canvas>

```

HitTestParameters Class

Definition: This is the base class for packing together parameters for a hit test pass.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

HitTestResult Class

Definition: This base returns the visual that was hit during a hit test pass.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

HwndInterop Class

Definition: HwndInterop

Method	Description
AttachChildWindow	AttachChildWindow
BeginPaint	BeginVisualPaint
EndPaint	EndVisualPaint
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.

GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetHwndParent	GetHwndParent
GetType	Gets the Type of the current instance. Inherited from Object.
HwndInterop	
Init	InitHwndSupport
InitWindow	InitWindow
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Shutdown	ShutdownHwndSupport
ToString	Returns a String that represents the current Object. Inherited from Object.

HwndVisual Class

Method	Description
ClearValue	Clears the local value of a property Inherited from DependencyObject.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Releases all resources held by the Visual object. Inherited from Visual.
FindCommonVisualAncestor	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetHwndVisual	GetHwndVisual
GetLocalValueEnumerator	Create a local value enumerator for this instance Inherited from DependencyObject.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Retrieve the value of a property Inherited from DependencyObject.
HitTest	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
HitTestCore	HitTestCore implements whether we have hit the bounds of this visual. Inherited from Visual.
InvalidateProperty	Invalidates a property Inherited from DependencyObject.
IsAncestorOf	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
IsDescendantOf	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
IVisual.FindCommonVisualAncestor	Inherited from Visual.
IVisual.HitTest	Inherited from Visual.
IVisual.IsAncestorOf	Inherited from Visual.
IVisual.IsDescendantOf	Inherited from Visual.
IVisual.TransformFromAncestor	Inherited from Visual.

IVisual.TransformFromDescendant	Inherited from Visual.
IVisual.TransformFromVisual	Inherited from Visual.
IVisual.TransformToAncestor	Inherited from Visual.
IVisual.TransformToDescendant	Inherited from Visual.
IVisual.TransformToVisual	Inherited from Visual.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
OnDelayedInvalidate	TODO: Left over from WCP FastBuild, determine relevance in future version of FastBuild Inherited from DependencyObject.
OnPropertyInvalidated	Notification that a specified property has been invalidated Inherited from DependencyObject.
ReadLocalValue	Retrieve the local value of a property (if set) Inherited from DependencyObject.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
SetValue	Sets the local value of a property Inherited from DependencyObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
TransformFromAncestor	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
TransformFromDescendant	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
TransformFromVisual	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
TransformToAncestor	Returns a Matrix object that represents the aggregate transformation from the coordinates of a Visual to the specified ancestor. Inherited from ContainerVisual.
TransformToDescendant	Returns a transformation matrix that can be used to transform coordinates from this node to a specified descendant Visual. Inherited from ContainerVisual.
TransformToVisual	Re-exposes the Visual base class's corresponding IVisual implementation as public method. Inherited from ContainerVisual.
ValidateProperty	Retrieve the value of a property (for use by native cache backed custom get accessors) Inherited from DependencyObject.
ValidatePropertyCore	Allows subclasses to participate in property value computation Inherited from DependencyObject.

Property	Description
Children	Gets a collection of the ContainerVisual's children. Inherited from ContainerVisual.
Clip	Gets or sets the clipping region of this ContainerVisual. Inherited from ContainerVisual.
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
DependencyObjectType	Returns the DType that represents the CLR type of this instance Inherited from DependencyObject.

Handle	Handle
HasChildren	Gets a value that indicates whether the ContainerVisual has a child collection. Inherited from ContainerVisual.
HitTestBounds	HitBounds returns the hit region bounding box for the current visual. Inherited from Visual.
IsDisposed	Gets a value that indicates whether the system has disposed of the Visual. Inherited from Visual.
IsHwndDpiAware	IsHwndDpiAware
Opacity	Gets or sets the opacity of the ContainerVisual. Inherited from ContainerVisual.
Parent	Gets the parent Visual. Inherited from ContainerVisual.
Size	Size

HyphenationCandidate Class

Definition: Describes one Hyphenation candidate.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
HyphenationCandidate	Ctor to create a new Hyphenation candidate
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
First	Get the first character to be inserted (if exist)
Index	Get the zero-based index of hyphenation position
Rule	Get the rule of this hyphenation
Second	Get the second character to be inserted (if exist)

ICCProfile Class

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
FixEndian	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetICCSignature	

GetType	Gets the Type of the current instance. Inherited from Object.
ICCSigned1516	
ICCWriteSigned1516	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetICCSignature	
ToString	Returns a String that represents the current Object. Inherited from Object.

ImageBrush Class

Definition: Fills an area with an image. This class may be used to specify images as the fill or background of other objects.

Method	Description
CloneCore	Implementation of Animatable.CloneCore.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable.
Copy	Creates a copy of this TileBrush. Inherited from TileBrush.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	Inherited from Brush.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Brush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time. That is, the returned brush is a snapshot of the current object at

	the point in time at which this method was called.
GetCurrentValue	Returns a non-animated version of this TileBrush that represents its current state. Inherited from TileBrush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageBrush	Initializes a new instance of the ImageBrush class.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from Brush.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method

should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
ContentUnits	Gets or sets a BrushMappingMode enumeration that specifies whether the value of the brush's ViewBox—the size and position of the brush's content—is relative to the size of the output area. This property only has an effect when the size of the brush's ViewPort is set to Rect.Empty. Inherited from TileBrush.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the brush has animations. Inherited from TileBrush.
HorizontalAlignment	Gets or sets a HorizontalAlignment enumeration that specifies how the brush's content is horizontally aligned within its tiles. Inherited from TileBrush.
ImageSource	Gets or sets an ImageSource object that contains the image.
IsAnimating	Inherited from TileBrush.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Inherited from TileBrush.
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.
SizeViewBoxToContent	Gets a Boolean that indicates whether the image is stretched to fill the entire output tile.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Stretch	Gets or sets a Stretch enumeration that specifies how the brush's selected content (ViewBox) is displayed in the brush's tiles (ViewPort). Inherited from TileBrush.
TileMode	Gets or sets a TileMode structure that specifies how the brush's tiles fill the output area. Inherited from TileBrush.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for

	maintaining thread safety. Inherited from <i>Changeable</i> .
VerticalAlignment	Gets or sets a <i>VerticalAlignment</i> enumeration that specifies how the brush's content is vertically aligned within its tiles. Inherited from <i>TileBrush</i> .
ViewBox	Gets or sets the position and dimensions of the brush's content. Inherited from <i>TileBrush</i> .
ViewBoxAnimations	Gets or sets a collection of <i>RectModifier</i> objects that animate the <i>ViewBox</i> of the brush. Inherited from <i>TileBrush</i> .
ViewPort	Gets or sets the position and dimensions of the brush's tiles. Inherited from <i>TileBrush</i> .
ViewPortAnimations	Gets or sets a collection of <i>RectModifier</i> objects that animate the <i>ViewPort</i> of the brush. Inherited from <i>TileBrush</i> .
ViewPortUnits	Gets or sets a <i>BrushMappingMode</i> enumeration that specifies whether the value of the brush's <i>ViewPort</i> —the size and position of the brush's tiles—is relative to the size of the output area. Inherited from <i>TileBrush</i> .

ImageCodecCollection Class

Definition: The collection of codecs (actually *CodecInfos*) on the system.

Method	Description
CopyTo	Copies the entire ImageCodecCollection to a compatible one-dimensional Array, starting at the specified index of the target array.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	Returns an enumerator to iterate through the codecs.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Codecs	Get the collection of codecs.
Count	Get count of codecs.
IsSynchronized	Whether this is thread-safe
Item	Indexer for returning a specific codec from the collection. The index must be in the range: (Count > codecIndex >= 0)
SyncRoot	Get an object that can be used to synchronize access

ImageC decEnumerator Class

Definition: The enumerator for Image frames.

Meth d	Description
CheckValidity	Checks for validity
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
MoveNext	MoveNext - Move to the next object in the collection. Returns false if the enumerator has passed the end of the collection
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Reset	Reset - resets the position to before the first object in the collection. A call to MoveNext must precede any call to Current after a Reset.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Current	Current - returns the current object in the collection

ImageColorTransform Class

Definition: ImageColorTransform Performs color management on an imaging pipeline.

Method	Description
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorContextFromPixelFormat	Given a pixel format, this function will return the closest standard color space (sRGB, scRGB, etc)
Copy	Overridden copy method for the output pin. Its job is to pull the bits from the Input through the color transform
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	

GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.
GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	This function returns the internal bitmap source for the output pin
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from ImageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageColorTransform	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Input	This is the first input, and is an alias for Inputs[0]
Inputs	This is the collection of inputs Inherited from ImageEffect.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Output	This is the first output, and is an alias for Outputs[0] Inherited from ImageEffect.
Outputs	This is the collection of outputs Inherited from ImageEffect.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ImageData Class

Definition: Contains an image and related data.

Method	Description
CanConvertTo	Whether the ImageSource can convert its data to the specified format. If not, a format converter could be used to do the conversion. Note: for best performance, ImageSources should provide support for PixelFormat32bppPARGB.
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Copies the pixel data from the image into an array of pixels. Inherited from ImageSource.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Copy a rect of pixel data from the image into the array of pixels that has the specified stride, starting at the pixelOffset (specified in number of pixels from the beginning). The pixels should be copied into the specified pixelFormat. To find out if the pixelFormat is supported, call ClosestPixelFormat first. An empty rect (one with a width and/or height of 0), means to ignore the rect and copy the entire image.
Create	Get the source string and any source properties from the imageData string and use them to construct an ImageData object.

EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Destroys resources associated with the object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetThumbnail	Returns a thumbnail of the image. Inherited from ImageSource.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageData	Initializes a new instance of the ImageData class.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from

	Changeable.
DecoderInfo	Get the information about the specific codec that was used to decode the image (if there a codec was required and we have that information).
DpiX	Get the horizontal dpi for the Image.
DpiY	Get the vertical dpi for the Image.
EmbeddedColorProfile	Gets the embedded color profile if one exists. Inherited from ImageSource.
EmbeddedThumbnail	If there is an embedded thumbnail for the Image, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one.
Format	Get the pixel format for the Image.
Height	Gets the height of the image in measure units (1/96 of an inch). Inherited from ImageSource.
InternalBitmapSource	Returns an IMILBitmapSource interface, if there is one, and AddRefs it.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
MetaData	Provides read-only access to this image's metadata.
Null	Gets an empty image source, i.e., an image source with 0 width and 0 height. Inherited from ImageSource.
Palette	Get/Sets a palette
PixelHeight	Get the pixel Height for the Image.
PixelWidth	Get the pixel width for the Image.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Width	Gets the width of the image in measure units (1/96 of an inch). Inherited from ImageSource.

ImageDataBuilder Class

Definition: This object is used to build an ImageData object.

Method	Description
Clear	Reset the builder to its default state
CreateInstance	Override from Builder Class
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageDataBuilder	Initializes a new instance of the ImageDataBuilder class.

MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToImageData	Create an ImageData from the current ImageDataBuilder properties.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
CreateCache	Whether to cache the image (not used for HBitmap and HIcon ctors).
DecoderInfo	The decoder to use to decode the image. If null, let the system decide which one to use.
DpiX	Get the horizontal dpi for the Image.
DpiY	Get the vertical dpi for the Image.
Format	Get the pixel format for the image.
ImageStream	The ImageStream property is used to specify where to get the image data from.
MetaData	Provides read-only access to this image's metadata.
PixelHeight	Get the pixel Height for the Image.
PixelWidth	Get the pixel width for the Image.
SizeOptions	How to size the image (if we're creating a cache).
Source	The source URI of the image to load.
SourceRect	Get/Set the source rect of the image. The source rect is only used when creating an image from a stream, a URI/filename, or from another ImageData. Other ways of creating an image ignore the source rect.
Uri	The Uri property is used to specify where to get the image data from.
UseEmbeddedColorProfile	Use an embedded color profile

ImageDecoder Class

Definition: ImageDecoder is a container for image frames. Each image frame is an ImageSource. Unlike ImageSource, ImageDecoder is NOT an immutable object and can be re-initialized to a different image stream. However, any ImageSources (frames) that it returns must be immutable.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Finalizer for ImageDecoder
GetEnumerator	Returns an enumerator to iterate through the frames of the image.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerable.GetEnumerator	
ImageDecoder	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first

	frame.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ...
DpiX	Get the horizontal dpi for the first frame.
DpiY	Get the vertical dpi for the first frame.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one.
Format	Get the pixel format for the first frame.
Height	Get the height of the image in measure units (96ths of an inch).
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe).
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0)
MetaData	Provides read-only access to this image's metadata.
PixelHeight	Get the pixel Height for the first frame.
PixelWidth	Get the pixel width for the first frame.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection.
Width	Get the width of the image in measure units (96ths of an inch).

ImageDecoderBmp Class

Definition: The built-in Microsoft Bmp (Bitmap) Decoder.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index. Inherited from ImageDecoderInternal.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	ImageDecoderInternal Finalizer. Inherited from ImageDecoderInternal.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied

	to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoderInternal.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerable.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderBmp	If this decoder cannot handle the image stream, it will throw an exception.
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream. Inherited from ImageDecoderInternal.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first frame. Inherited from ImageDecoder.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image. Inherited from ImageDecoderInternal.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame. Inherited from ImageDecoderInternal.
DpiY	Get the vertical dpi for the first frame. Inherited from ImageDecoderInternal.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. Inherited from ImageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one. Inherited from ImageDecoderInternal.
Format	Get the pixel format for the first frame. Inherited from ImageDecoderInternal.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe). Inherited from ImageDecoderInternal.
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0) Inherited from ImageDecoderInternal.
MetaData	Provides read-only access to this image's metadata. Inherited from ImageDecoderInternal.
PixelHeight	Get the pixel Height for the first frame. Inherited from ImageDecoderInternal.
PixelWidth	Get the pixel width for the first frame. Inherited from ImageDecoderInternal.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection. Inherited from ImageDecoderInternal.
Width	Get the width of the image in measure units (96ths of an inch). Inherited from

ImageDecoder.

ImageDecoderGif Class

Definition: The built-in Microsoft GIF Decoder.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index. Inherited from ImageDecoderInternal.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	ImageDecoderInternal Finalizer. Inherited from ImageDecoderInternal.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoderInternal.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerable.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderGif	If this decoder cannot handle the image stream, it will throw an exception.
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream. Inherited from ImageDecoderInternal.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first frame. Inherited from ImageDecoder.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image. Inherited from ImageDecoderInternal.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame. Inherited from ImageDecoderInternal.
DpiY	Get the vertical dpi for the first frame. Inherited from ImageDecoderInternal.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. Inherited from ImageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise,

	return null. This method does NOT create a thumbnail for images that don't already have one. Inherited from ImageDecoderInternal.
Format	Get the pixel format for the first frame. Inherited from ImageDecoderInternal.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe). Inherited from ImageDecoderInternal.
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0) Inherited from ImageDecoderInternal.
MetaData	Provides read-only access to this image's metadata. Inherited from ImageDecoderInternal.
PixelHeight	Get the pixel Height for the first frame. Inherited from ImageDecoderInternal.
PixelWidth	Get the pixel width for the first frame. Inherited from ImageDecoderInternal.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection. Inherited from ImageDecoderInternal.
Width	Get the width of the image in measure units (96ths of an inch). Inherited from ImageDecoder.

ImageDecoderIcon Class

Definition: The built-in Microsoft Icon Decoder.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index. Inherited from ImageDecoderInternal.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	ImageDecoderInternal Finalizer. Inherited from ImageDecoderInternal.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoderInternal.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerator.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderIcon	If this decoder cannot handle the image stream, it will throw an exception.
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream. Inherited from ImageDecoderInternal.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first

ReferenceEquals	frame. Inherited from ImageDecoder. Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image. Inherited from ImageDecoderInternal.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame. Inherited from ImageDecoderInternal.
DpiY	Get the vertical dpi for the first frame. Inherited from ImageDecoderInternal.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. Inherited from ImageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one. Inherited from ImageDecoderInternal.
Format	Get the pixel format for the first frame. Inherited from ImageDecoderInternal.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe). Inherited from ImageDecoderInternal.
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0) Inherited from ImageDecoderInternal.
MetaData	Provides read-only access to this image's metadata. Inherited from ImageDecoderInternal.
PixelHeight	Get the pixel Height for the first frame. Inherited from ImageDecoderInternal.
PixelWidth	Get the pixel width for the first frame. Inherited from ImageDecoderInternal.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection. Inherited from ImageDecoderInternal.
Width	Get the width of the image in measure units (96ths of an inch). Inherited from ImageDecoder.

ImageDecoderInternal Class

Definition: For internal use only.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Image DecoderInternal Finalizer.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty

	rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerator.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderInternal	
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first frame. Inherited from ImageDecoder.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame.
DpiY	Get the vertical dpi for the first frame.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. Inherited from ImageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one.
Format	Get the pixel format for the first frame.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder). Inherited from ImageDecoder.
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe).
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0)
MetaData	Provides read-only access to this image's metadata.
PixelHeight	Get the pixel Height for the first frame.
PixelWidth	Get the pixel width for the first frame.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection.
Width	Get the width of the image in measure units (96ths of an inch). Inherited from ImageDecoder.

ImageDecoderJpeg Class

Definition: The built-in Microsoft Jpeg Decoder.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index. Inherited from ImageDecoderInternal.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	ImageDecoderInternal Finalizer. Inherited from ImageDecoderInternal.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from imageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoderInternal.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerable.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderJpeg	If this decoder cannot handle the image stream, it will throw an exception.
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream. Inherited from ImageDecoderInternal.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first frame. Inherited from ImageDecoder.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image. Inherited from ImageDecoderInternal.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame. Inherited from ImageDecoderInternal.
DpiY	Get the vertical dpi for the first frame. Inherited from ImageDecoderInternal.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. Inherited from imageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one. Inherited from ImageDecoderInternal.

Format	Get the pixel format for the first frame. Inherited from ImageDecoderInternal.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe). Inherited from ImageDecoderInternal.
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0) Inherited from ImageDecoderInternal.
MetaData	Provides read-only access to this image's metadata. Inherited from ImageDecoderInternal.
PixelHeight	Get the pixel Height for the first frame. Inherited from ImageDecoderInternal.
PixelWidth	Get the pixel width for the first frame. Inherited from ImageDecoderInternal.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection. Inherited from ImageDecoderInternal.
Width	Get the width of the image in measure units (96ths of an inch). Inherited from ImageDecoder.

ImageDecoderPng Class

Definition: The built-in Microsoft PNG Decoder.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index. Inherited from ImageDecoderInternal.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	ImageDecoderInternal Finalizer. Inherited from ImageDecoderInternal.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoderInternal.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerator.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderPng	If this decoder cannot handle the image stream, it will throw an exception.
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream. Inherited from ImageDecoderInternal.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first frame. Inherited from ImageDecoder.
ReferenceEquals	Determines whether the specified Object instances are the same

instance. Inherited from Object.	
ToString	Returns a String that represents the current Object. Inherited from Object.
Property	Description
Count	The number of image frames in this image. Inherited from ImageDecoderInternal.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame. Inherited from ImageDecoderInternal.
DpiY	Get the vertical dpi for the first frame. Inherited from ImageDecoderInternal.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. inherited from imageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one. Inherited from ImageDecoderInternal.
Format	Get the pixel format for the first frame. Inherited from ImageDecoderInternal.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe). Inherited from ImageDecoderInternal.
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0) Inherited from ImageDecoderInternal.
MetaData	Provides read-only access to this image's metadata. Inherited from ImageDecoderInternal.
PixelHeight	Get the pixel Height for the first frame. Inherited from ImageDecoderInternal.
PixelWidth	Get the pixel width for the first frame. Inherited from ImageDecoderInternal.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection. Inherited from ImageDecoderInternal.
Width	Get the width of the image in measure units (96ths of an inch). Inherited from ImageDecoder.

ImageDecoderTiff Class

Definition: The built-in Microsoft Tiff Decoder.

Method	Description
CopyTo	Copies the frames to an Array, starting at a particular Array index. Inherited from ImageDecoderInternal.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	ImageDecoderInternal Finalizer. Inherited from ImageDecoderInternal.
GetEnumerator	Returns an enumerator to iterate through the frames of the image. Inherited from ImageDecoder.
GetFrame	Return an image frame, cropped by the specified sourceRect. An empty rect (one with a width and/or height of 0), means to ignore the rect and get

	the entire frame. Inherited from ImageDecoder.
GetFrame	Return an image frame, with all the specified cropping and sizing applied to it. An empty source rect (one with a width and/or height of 0), means to ignore the rect and get the entire frame. Inherited from ImageDecoderInternal.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IEnumerable.GetEnumerator	Inherited from ImageDecoder.
ImageDecoderTiff	If this decoder cannot handle the image stream, it will throw an exception.
Initialize	Initialize the codec. Instantiate the correct unmanaged codec by using the supplied Guid and then initialize it to the stream. Inherited from ImageDecoderInternal.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
op_Explicit	Conversion from ImageDecoder to ImageSource by returning the first frame. Inherited from ImageDecoder.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	The number of image frames in this image. Inherited from ImageDecoderInternal.
CustomEncoderProperties	Returns a codec-specific object that identifies the properties that were used to encode this image. These codec-specific properties can be passed back to the associated encoder (if there is one) to get the same type of encoding again. This is approximate ... Inherited from ImageDecoder.
DpiX	Get the horizontal dpi for the first frame. Inherited from ImageDecoderInternal.
DpiY	Get the vertical dpi for the first frame. Inherited from ImageDecoderInternal.
EmbeddedColorProfile	If there is an embedded color profile, return it. Otherwise, return null. This method does NOT create a color profile for images that don't already have one. Inherited from ImageDecoder.
EmbeddedThumbnail	If there is an embedded thumbnail for the first frame, return it. Otherwise, return null. This method does NOT create a thumbnail for images that don't already have one. Inherited from ImageDecoderInternal.
Format	Get the pixel format for the first frame. Inherited from ImageDecoderInternal.
Height	Get the height of the image in measure units (96ths of an inch). Inherited from ImageDecoder.
Info	The info that identifies this codec (including any associated encoder).
IsSynchronized	A value indicating whether access to the ICollection is synchronized (thread-safe). Inherited from ImageDecoderInternal.
Item	Indexer for returning a specific frame of the image (at full size). The index must be in the range: (NumFrames > index >= 0) Inherited from ImageDecoderInternal.
MetaData	Provides read-only access to this image's metadata. Inherited from ImageDecoderInternal.
PixelHeight	Get the pixel Height for the first frame. Inherited from ImageDecoderInternal.
PixelWidth	Get the pixel width for the first frame. Inherited from ImageDecoderInternal.
SyncRoot	Gets an object that can be used to synchronize access to the

ICollection. Inherited from ImageDecoderInternal.

Width

Get the width of the image in measure units (96ths of an inch). Inherited from ImageDecoder.

ImageEffect Class

Definition: The ImageEffect class is the base class for all imaging effects (blur, grayscale, etc) It's possible for an effect to not have any inputs but an effect must always have at least one output. The default implementations of things assume this. If a derived effect is going to play with Output/Outputs be sure that at least one is there.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	ImageSource abstract method implementation PixelOffset actually doesn't do anything. If you don't want to start at (0,0) in the input, then have your sourceRect start at the point you want.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. inherited from Object.
GetDpiX	Horizontal DPI of the image.
GetDpiY	Vertical DPI of the image.
GetFormat	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	ImageSource abstract method implementation
GetOutput	
GetPalette	Get a palette for a particular output
GetPixelHeight	Height, in pixels, of the image.
GetPixelWidth	Width, in pixels, of the image.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current,

logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space.

GetScaleY	
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffect	This constructor allows a derived class to specify the number of inputs and outputs for this effect, and this class will then handle the input and output arrays, including validation of indices. This defaults to 1 and 1. If the effect wishes to have a variable number of inputs or outputs, it can pass -1 for either (or both) counts, and the input and output collections will allow this. Finally, these methods are all virtual, so derived classes may choose not to delegate back to the base class, in which case no extra cost is incurred.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Input	This is the first input, and is an alias for Inputs[0] Performance

	Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you.
Inputs	This is the collection of inputs
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Output	This is the first output, and is an alias for Outputs[0]
Outputs	This is the collection of outputs
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ImageEffectBlur Class

Definition: Gaussian blur effect. It is a single input, single output effect. Warning: If the effect is being scaled (i.e. input.ScaleX or input.ScaleY isn't 1) and Expand is true, then it's possible for the output dimensions to be larger or smaller than PixelWidth and PixelHeight. Adjust the pixel buffer fed to copy to avoid problems.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.
GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.

GetInternalBitmapSource	Required by ImageEffect
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and imageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from imageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectBlur	Default Constructor
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.

CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Expand	If true, the blur will spread out and create a larger image. If false, the blur will be contained.
Input	This is the first input, and is an alias for Inputs[0] Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from ImageEffect.
Inputs	This is the collection of inputs Inherited from ImageEffect.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Output	This is the first output, and is an alias for Outputs[0] Inherited from ImageEffect.
Outputs	This is the collection of outputs Inherited from ImageEffect.
Radius	Radius of the blur kernel.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UiContext	Gets the UiContext of the current object. The UiContext is used for maintaining thread safety. Inherited from Changeable.

ImageEffectFlipRotate Class

Definition: This effect can flip an image in X or Y and rotate by multiples of 90 deg.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.

GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	Required by ImageEffect
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from ImageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectFlipRotate	FlipRotate constructor. It is a single input, single output effect.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property

Description

AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
FlipX	Flip the image over the horizontal axis
FlipY	Flip the image over the vertical axis
Input	This is the first input, and is an alias for Inputs[0] Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from ImageEffect.
Inputs	This is the collection of inputs Inherited from ImageEffect.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Output	This is the first output, and is an alias for Outputs[0] Inherited from ImageEffect.
Outputs	This is the collection of outputs Inherited from ImageEffect.
Rotation	The number of degrees to rotate. This must be a multiple of 90 degrees. A positive number rotates CW and a negative number rotates CCW.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ImageEffectGammaCorrect Class

Definition: This effect changes the gamma of an image.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from

	Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.
GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	Required by ImageEffect
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from ImageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectGammaCorrect	Gamma correction effect. It is a single input, single output effect.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the

WritePreamble	OnChanged method. Inherited from Changeable. Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.
---------------	--

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
BlueGamma	The correction for blue (1.0 means no change)
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
GreenGamma	The correction for green (1.0 means no change)
Input	This is the first input, and is an alias for Inputs[0] Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from ImageEffect.
Inputs	This is the collection of inputs Inherited from ImageEffect.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Output	This is the first output, and is an alias for Outputs[0] Inherited from ImageEffect.
Outputs	This is the collection of outputs Inherited from ImageEffect.
RedGamma	The correction for red (1.0 means no change)
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ImageEffectGlow Class

Definition: Performs a glow effect. It is a single input, single output effect.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be

	retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.
GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	Required by ImageEffect
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from ImageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectGlow	Default Constructor
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AspectRatio	The aspect ratio of the glow (i.e. it impacts how much it glows in one direction over another)
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Composite	If this is true, the source image is pasted on top of the glow. If this is false, all the effect does is return the glow.
InnerColor	The inner of the two glow colors.
Input	This is the first input, and is an alias for Inputs[0] Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from ImageEffect.
Inputs	This is the collection of inputs Inherited from ImageEffect.
Intensity	The glow intensity.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
OuterColor	The outer of the two glow colors.
Output	This is the first output, and is an alias for Outputs[0] Inherited from ImageEffect.
Outputs	This is the collection of outputs Inherited from ImageEffect.
Size	Size of the glow.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ImageEffectGrayscale Class

Definition: Converts an image to grayscale. It is a single input, single output effect.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from

	Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
EmbeddedChangeableReader	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
EmbeddedChangeableWriter	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
Equals	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
Finalize	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
GetDpiX	Horizontal DPI of the image. Inherited from <code>ImageEffect</code> .
GetDpiY	Vertical DPI of the image. Inherited from <code>ImageEffect</code> .
GetFormat	Inherited from <code>ImageEffect</code> .
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
GetInternalBitmapSource	Required by <code>ImageEffect</code>
GetOutput	Inherited from <code>ImageEffect</code> .
GetPalette	Get a palette for a particular output Inherited from <code>ImageEffect</code> .
GetPixelHeight	Height, in pixels, of the image. Inherited from <code>ImageEffect</code> .
GetPixelWidth	Width, in pixels, of the image. Inherited from <code>ImageEffect</code> .
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and <code>ImageEffectSource</code> space. Inherited from <code>ImageEffect</code> .
GetScaleY	Inherited from <code>ImageEffect</code> .
GetType	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
ImageEffectGrayscale	Default Constructor
MakeUnchangeable	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
MakeUnchangeableCore	Makes a <code>Changeable</code> object immutable. Inherited from <code>Changeable</code> .
MemberwiseClone	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
ModifyHandlerIfChangeable	Adds or removes a <code>Changed</code> event handler to or from the specified <code>Changeable</code> object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
OnChanged	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .

PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Input	This is the first input, and is an alias for Inputs[0] Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from ImageEffect.
inputs	This is the collection of inputs Inherited from ImageEffect.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Output	This is the first output, and is an alias for Outputs[0] Inherited from ImageEffect.
Outputs	This is the collection of outputs Inherited from ImageEffect.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ImageEffectNegate Class

Definition: Negates an image. It is a single input, single output effect.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.
GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	Required by ImageEffect
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from ImageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectNegate	Default Constructor
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified

	object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
<code>OnChanged</code>	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
<code>PropagateEventHandler</code>	Shares a <code>Changed</code> event handler with the current object's data members or removes it. Inherited from <code>Changeable</code> .
<code>ReadPreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
<code>ReferenceEquals</code>	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
<code>ToString</code>	Returns a <code>String</code> that represents the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ValidateObjectState</code>	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
<code>WritePostscript</code>	Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code> .
<code>WritePreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from <code>Changeable</code> .

Property	Description
<code>AllowChangeableReferenceOverride</code>	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>Input</code>	This is the first input, and is an alias for <code>Inputs[0]</code> Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from <code>ImageEffect</code> .
<code>Inputs</code>	This is the collection of inputs Inherited from <code>ImageEffect</code> .
<code>IsChangeable</code>	Gets a <code>Boolean</code> that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
<code>Output</code>	This is the first output, and is an alias for <code>Outputs[0]</code> Inherited from <code>ImageEffect</code> .
<code>Outputs</code>	This is the collection of outputs Inherited from <code>ImageEffect</code> .
<code>StatusOfNextUse</code>	Gets or sets a <code>UseStatus</code> enumeration that specifies how the <code>Changeable</code> object behaves when it is "used." A <code>Changeable</code> object is considered used in the following situations: the object is set into a <code>Property System</code> property, the object is used as a sub-object in a complex <code>Changeable</code> object, or the object is used in a <code>DrawingContext</code> command. Inherited from <code>Changeable</code> .
<code>UIContext</code>	Gets the <code>UIContext</code> of the current object. The <code>UIContext</code> is used for maintaining thread safety. Inherited from <code>Changeable</code> .

ImageEffectSharpen Class

Definition: Unsharp mask. It is a single input, single output effect.

Method	Description
CloneCore	Required by changeable?
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	This applies the effect and places the result in "pixels"
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetDpiX	Horizontal DPI of the image. Inherited from ImageEffect.
GetDpiY	Vertical DPI of the image. Inherited from ImageEffect.
GetFormat	Inherited from ImageEffect.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetInternalBitmapSource	Required by ImageEffect
GetOutput	Inherited from ImageEffect.
GetPalette	Get a palette for a particular output Inherited from ImageEffect.
GetPixelHeight	Height, in pixels, of the image. Inherited from ImageEffect.
GetPixelWidth	Width, in pixels, of the image. Inherited from ImageEffect.
GetScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space. Inherited from ImageEffect.
GetScaleY	Inherited from ImageEffect.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectSharpen	Default Constructor
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified

	object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
<code>OnChanged</code>	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
<code>PropagateEventHandler</code>	Shares a <code>Changed</code> event handler with the current object's data members or removes it. Inherited from <code>Changeable</code> .
<code>ReadPreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
<code>ReferenceEquals</code>	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
<code>ToString</code>	Returns a <code>String</code> that represents the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ValidateObjectState</code>	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
<code>WritePostscript</code>	Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code> .
<code>WritePreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from <code>Changeable</code> .

Property	Description
<code>AllowChangeableReferenceOverride</code>	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to "ChangeableReference". Inherited from <code>Changeable</code> .
<code>Amount</code>	How much to sharpen
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>Input</code>	This is the first input, and is an alias for <code>Inputs[0]</code> Performance Warning: If the input of the effect IS NOT in a format that the effect supports the effect will convert the input to a workable format for you. Inherited from <code>ImageEffect</code> .
<code>Inputs</code>	This is the collection of inputs Inherited from <code>ImageEffect</code> .
<code>IsChangeable</code>	Gets a <code>Boolean</code> that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
<code>Output</code>	This is the first output, and is an alias for <code>Outputs[0]</code> Inherited from <code>ImageEffect</code> .
<code>Outputs</code>	This is the collection of outputs Inherited from <code>ImageEffect</code> .
<code>Radius</code>	The radius of the blur performed before sharpening
<code>StatusOfNextUse</code>	Gets or sets a <code>UseStatus</code> enumeration that specifies how the <code>Changeable</code> object behaves when it is "used." A <code>Changeable</code> object is considered used in the following situations: the object is set into a <code>Property System</code> property, the object is used as a sub-object in a complex <code>Changeable</code> object, or the object is used in a <code>DrawingContext</code> command. Inherited from <code>Changeable</code> .
<code>UIContext</code>	Gets the <code>UIContext</code> of the current object. The <code>UIContext</code> is used for

maintaining thread safety. Inherited from Changeable.

ImageEffectSource Class

Definition: ImageEffectSource class implementation.

Method	Description
CanConvertTo	Gets or sets a value that indicates whether the image source can convert its data to the specified format. Inherited from ImageSource.
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Copies the pixel data from the image into an array of pixels. Inherited from ImageSource.
Copy	ImageSource abstract method implementation
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from ImageSource.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetThumbnail	Returns a thumbnail of the image. Inherited from ImageSource.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageEffectSource	Constructor
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.

ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DpiX	Horizontal DPI of the image.
DpiY	Vertical DPI of the image.
EmbeddedColorProfile	Gets the embedded color profile if one exists. Inherited from ImageSource.
EmbeddedThumbnail	Gets the embedded thumbnail if one exists. Inherited from ImageSource.
Format	
Height	Gets the height of the image in measure units (1/96 of an inch). Inherited from ImageSource.
ImageEffect	This provides access to the effect from which this source came.
InternalBitmapSource	ImageSource abstract method implementation
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
MetaData	Gets the metadata of the image. Inherited from ImageSource.
Null	Gets an empty image source, i.e., an image source with 0 width and 0 height. Inherited from ImageSource.
Palette	Get a palette
PixelHeight	Height, in pixels, of the image.
PixelWidth	Width, in pixels, of the image.
ScaleX	These values contain the horizontal and vertical scale applied to this source. There are occasions when an effect needs to operate at a different resolution or a different coordinate space than the current, logical coordinate space. Thus, these properties enable the consumer to map between local space and ImageEffectSource space.
ScaleY	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the

Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

UIContext Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Width Gets the width of the image in measure units (1/96 of an inch). Inherited from imageSource.

ImageEffectSourceCollection Class

Definition: The collection of image effect outputs.

Method	Description
CopyTo	Copies the entire ImageEffectSourceCollection to a compatible one-dimensional Array, starting at the specified index of the target array.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	Returns an enumerator to iterate through the outputs
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
MoveNext	Moves the enumerator to the next element in the collection; returns false if beyond end.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Reset	Moves the enumerator to the next element in the collection; returns false if beyond end.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	Get count of outputs.
Current	Returns the current element in the enumeration
IsSynchronized	Returns whether the object is thread-safe
Item	Indexer for returning a specific image effect output from the collection. The index must be in the range: (Count > maxOutputs >= 0)
SyncRoot	Returns the underlying root object for which synchronization occurs.

ImageSourceConverter Class

Definition: ImageSourceConverter

Method	Description
CanConvertFrom	CanConvertFrom
CanConvertTo	Inherited from TypeConverter.

CanConvertTo	Inherited from TypeConverter.
CanConvertTo	TypeConverter method override.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	ConvertFromString
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	TypeConverter method implementation.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
ImageSourceConverter	
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

IntegerCollection Class

Method	Description
Add	
AddRange	
Clear	
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	

Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
Copy	
CopyTo	
EmbeddedChangeableReader	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
EmbeddedChangeableWriter	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
Equals	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
Finalize	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
GetRange	
GetType	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
IntegerCollection	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
MakeUnchangeableCore	Makes a <code>Changeable</code> object immutable. Inherited from <code>Changeable</code> .
MemberwiseClone	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
ModifyHandlerIfChangeable	Adds or removes a <code>Changed</code> event handler to or from the specified <code>Changeable</code> object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
OnChanged	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
PropagateEventHandler	Shares a <code>Changed</code> event handler with the current object's data

	members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	
Count	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

IntegerCollecti nConverter Class

Definition: IntegerCollectionConverter - Converter class for converting instances of other types to and from IntegerCollection instances.

Meth d	Description
CanConvertFrom	Inherited from TypeConverter.
CanConvertFrom	CanConvertFrom - Returns whether or not this class can convert from a given type.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	CanConvertTo - Returns whether or not this class can convert to a given type.
ConvertFrom	ConvertFrom - Attempt to convert to a IntegerCollection from the given object
ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	ConvertTo - Attempt to convert a IntegerCollection to the given type
ConvertTo	Inherited from TypeConverter.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IntegerCollectionConverter	
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

LinearGradientBrush Class

Definition: Defines a linear gradient used to fill an area.

Method	Description
AddStop	Adds a gradient stop to the brush. Inherited from GradientBrush.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from GradientBrush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	Inherited from GradientBrush.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from GradientBrush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this GradientBrush that represents its current state. Inherited from GradientBrush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time. That is, the returned brush is a snapshot of the current object at the point in time at which this method was called.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in

	hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
LinearGradientBrush	initializes a new instance of the LinearGradientBrush class.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Inherited from GradientBrush.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from GradientBrush.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.

CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
ColorInterpolationMode	ColorInterpolationMode - Read only accessor of the ColorInterpolationMode property. This property controls how the colors in Gradient are interpolated. Default is ColorInterpolationMode.PerceptuallyLinearGamma. Inherited from GradientBrush.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
EndPoint	Gets or sets the ending two-dimensional coordinates of the linear gradient.
EndPointAnimations	Gets or sets a collection of PointModifier objects that animate the brush's EndPoint property.
GradientStops	Gets or sets the gradient stops (transition points) of a brush. Inherited from GradientBrush.
HasAnimations	Gets a Boolean that indicates whether the brush has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Gets a Boolean that indicates whether the LinearGradientBrush is currently animated.
MappingMode	Gets or sets a BrushMappingMode enumeration that specifies whether the gradient brush's positioning coordinates are absolute or relative to the output area. inherited from GradientBrush.
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.
SpreadMethod	Gets or sets the type of spread method that specifies how to draw a gradient that starts or ends inside the bounds of the object to be painted. inherited from GradientBrush.
StartPoint	Gets or sets the starting coordinates of the linear gradient.
StartPointAnimations	Gets or sets a collection of PointModifier objects that animate the brush's StartPoint property.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to use the Transform property of brushes to apply transformations to LinearGradientBrush and RadialGradientBrush fills. A LinearGradientBrush is used to fill the first two Rectangle elements. The difference between the rectangles is that the LinearGradientBrush in the second

rectangle is rotated 45 degrees. The second pair of rectangles illustrates the before and after effect of a ScaleTransform by reducing a RadialGradientBrush to half its normal height.

```
<Border xmlns="http://schemas.microsoft.com/2003/xaml"
Background="#CCCCCC">

<Canvas Height="40">

<!-- Rectangle #1 is filled with a LinearGradientBrush. The gradient colors
flow from left to right by default. -->

<Rectangle RectangleLeft="10" RectangleTop="10"
RectangleWidth="300" RectangleHeight="200">

<Rectangle.Fill>
<LinearGradientBrush>

<LinearGradientBrush.GradientStops>
<GradientStopCollection>
<GradientStop Color="red" Offset="0"/>
<GradientStop Color="yellow" Offset="1" />
<GradientStop Color="blue" Offset="0.5"/>
<GradientStop Color="white" Offset="0.2"/>
</GradientStopCollection>
</LinearGradientBrush.GradientStops>

</LinearGradientBrush>
</Rectangle.Fill>

</Rectangle>

<!-- Rectangle #2 is identical to the first rectangle except that the Transform
property rotates the LinearGradientBrush so that the gradient colors are
rotated by 45 degrees. -->

<Rectangle RectangleLeft="320" RectangleTop="10"
RectangleWidth="300" RectangleHeight="200">

<Rectangle.Fill>
<LinearGradientBrush>

<LinearGradientBrush.Transform>
<RotateTransform Angle="45" /> <!-- Rotation angle. -->
</LinearGradientBrush.Transform>

<LinearGradientBrush.GradientStops>
<GradientStopCollection>
<GradientStop Color="red" Offset="0"/>
<GradientStop Color="yellow" Offset="1" />
<GradientStop Color="blue" Offset="0.5"/>
<GradientStop Color="white" Offset="0.2"/>
</GradientStopCollection>
</LinearGradientBrush.GradientStops>

</LinearGradientBrush>
```

```

</Rectangle.Fill>

</Rectangle>

<!-- Rectangle #3 is filled with a RadialGradientBrush. -->

<Rectangle RectangleLeft="10" RectangleTop="250"
  RectangleWidth="300" RectangleHeight="200">

  <Rectangle.Fill>
    <RadialGradientBrush Focus="0.5,0.5">

      <RadialGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="red" Offset="0"/>
          <GradientStop Color="yellow" Offset="1"/>
          <GradientStop Color="blue" Offset="0.5"/>
        </GradientStopCollection>
      </RadialGradientBrush.GradientStops>

    </RadialGradientBrush>
  </Rectangle.Fill>

</Rectangle>

<!-- Rectangle #4 is identical to the third rectangle except that the Transform
  property applies a ScaleTransform to the RadialGradientBrush so that the
  gradient is half its previous height. -->

<Rectangle RectangleLeft="320" RectangleTop="250"
  RectangleWidth="300" RectangleHeight="200">

  <Rectangle.Fill>

    <RadialGradientBrush Focus="0.5,0.5">
      <RadialGradientBrush.Transform>
        <ScaleTransform ScaleX="1" ScaleY="0.5" /><!-- Scale transform. -->
      </RadialGradientBrush.Transform>

      <RadialGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="red" Offset="0"/>
          <GradientStop Color="yellow" Offset="1"/>
          <GradientStop Color="blue" Offset="0.5"/>
        </GradientStopCollection>
      </RadialGradientBrush.GradientStops>

    </RadialGradientBrush>
  </Rectangle.Fill>

</Rectangle>

</Canvas>
</Border>

```


This example creates simple vertical, horizontal, and radial gradients and uses them to fill an element using "Longhorn" markup language (code-named "XAML").

In the following example, vertical and horizontal gradients are used to set the Fill property of two Rectangle elements. In this particular example, the gradients are described using simple notation: GradientType StartColor EndColor, where GradientType is VerticalGradient, HorizontalGradient, or RadialGradient. StartColor and EndColor can be predefined color names (such as Blue) or hexadecimal values.

```
<Canvas xmlns="http://schemas.microsoft.com/2003/xaml">
```

```
<Rectangle
  Fill="VerticalGradient Blue Green"
  RectangleLeft="20"
  RectangleTop="20"
  RectangleWidth="100"
  RectangleHeight="100">
```

```
<Rectangle
  Fill="HorizontalGradient Blue Red"
  RectangleLeft="120"
  RectangleTop="120"
  RectangleWidth="100"
  RectangleHeight="100">
```

A vertical gradient is a linear gradient whose start and endpoints form a vertical line; likewise, a horizontal gradient is a linear gradient whose start and endpoints form a horizontal line. You can explicitly describe your own linear gradients using the following syntax:

LinearGradient StartPoint EndPoint StartColor EndColor, where StartPoint and EndPoint are the starting and ending coordinates, with each coordinate expressed as a pair of x and y values from 0 to 1, such as 0.1,0.1 and 0.5,0.5. These values indicate the relative position of the start or end point. An endpoint of 0.5,0.5 would be located 50 percent to the right of the fill area and 50 percent of the way from the top of the area—the middle of the shape.

In the following example, the Fill property of a Rectangle element is set by explicitly using a linear gradient.

```
<Rectangle
  Fill="LinearGradient 0.1,0.1 0.5,0.5 Blue Green"
  RectangleLeft="220"
  RectangleTop="220"
  RectangleWidth="100"
  RectangleHeight="100">
```

In the final example, the Fill property of a Rectangle element is set using a radial gradient.

```
<Rectangle
  Fill="RadialGradient Blue Red"
  RectangleLeft="320"
  RectangleTop="320"
  RectangleWidth="100"
  RectangleHeight="100">
```

</Rectangle>

</Canvas>

See [Create a Gradient with More Than Two Colors](#) for an example of how to create a gradient with more than two gradient stops.

To create vertical and horizontal gradients in code or using compound notation, use the `LinearGradientBrush` class and set its `StartPoint` and `EndPoint` properties so that they describe a vertical or horizontal line. To create radial gradients in code or using compound notation, use the `RadialGradientBrush` class.

This example demonstrates how to create a gradient that has more than two colors in "XAML". To create a gradient with more than two colors, add a `GradientStopCollection` to the gradient's `GradientStops` property. Next, add `GradientStop` objects to the `GradientStopCollection`, one for each color the gradient should contain. Set the `Color` and the `Offset`, a value from 0 to 1 that determines the relative position of the stop in the gradient, of each of the stops. The following example shows a `Button` whose `Background` is filled with a horizontal gradient that has four colors.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Button
    Canvas.Top="50"
    Canvas.Left="50"
    BorderBrush="Black"
    Width="200"
    Height="30">
    <Button.Background>
      <LinearGradientBrush >
        <LinearGradientBrush.GradientStops>
          <GradientStopCollection>
            <GradientStop Color="Red" Offset="0" />
            <GradientStop Color="Blue" Offset="0.25"/>
            <GradientStop Color="Orange" Offset="0.75"/>
            <GradientStop Color="Yellow" Offset="1"/>
          </GradientStopCollection>
        </LinearGradientBrush.GradientStops>
      </LinearGradientBrush>
    </Button.Background>
  </Button>

</Canvas>
```

LineGeometry Class

Definition: Represents the geometry of a line.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .
CloneCore	Implementation of <code>Animatable.CloneCore</code> .
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .

Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
Copy	Creates a copy of this <code>Geometry</code> . Inherited from <code>Geometry</code> .
Copy	Creates a copy of this <code>LineGeometry</code> .
DisableCore	Inherited from <code>Geometry</code> .
Dispose	Releases the resources associated with the object. Inherited from <code>Geometry</code> .
DoesContain	Returns if point is inside the geometry. Inherited from <code>Geometry</code> .
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from <code>Animatable</code> .
EmbeddedAnimationCollectionWriter	Inherited from <code>Animatable</code> .
EmbeddedChangeableReader	Inherited from <code>Animatable</code> .
EmbeddedChangeableWriter	Inherited from <code>Animatable</code> .
EnableCore	Inherited from <code>Geometry</code> .
Equals	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
Finalize	Inherited from <code>Geometry</code> .
GetBounds	Returns the bounds of a <code>Geometry</code> , widened according to the characteristics of the specified pen. Inherited from <code>Geometry</code> .
GetCurrentValue	Returns a non-animated version of this <code>LineGeometry</code> that represents its current state.
GetCurrentValue	Returns a non-animated version of this <code>Animatable</code> that represents its current state. Inherited from <code>Animatable</code> .
GetCurrentValue	Returns a non-animated version of this <code>Geometry</code> that represents its current state. Inherited from <code>Geometry</code> .
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
GetType	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
LineGeometry	Initializes a new instance of the <code>LineGeometry</code> class.
MakeUnchangeable	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
MakeUnchangeableCore	Inherited from <code>Geometry</code> .
MemberwiseClone	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
ModifyHandlerIfChangeable	Adds or removes a <code>Changed</code> event handler to or from the specified <code>Changeable</code> object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
OnChanged	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
PropagateEventHandler	Implementation of <code>PropagateEventHandler</code> .
ReadPreamble	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .

ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from Geometry.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
Bounds	Gets a Rect that specifies the bounding box of a LineGeometry. This method does not take any pens into account.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
EndPoint	Gets or sets the end point of a line.
EndPointAnimations	Gets or sets a collection of PointModifier objects that animate the line's EndPoint.
HasAnimations	Gets a Boolean that indicates whether the geometry has animations.
IsAnimating	Returns true if the geometry contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
StartPoint	Gets or sets the start point of a line.
StartPointAnimations	Gets or sets a collection of PointModifier objects that animate a line's StartPoint.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

Transform	Gets or sets the Transform object applied to a Geometry. Inherited from Geometry.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
```

```
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();
```

```
// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));
```

```
' VB .NET
```

```
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

```
' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))
```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
```

```
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));
```

```
myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));
```

```
myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));
```

```
MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);
```

```
myPathFigure.AddSegment(myBezierSegment);
```

```
myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))
```

```
Dim myBezierSegment As new BezierSegment(_
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(200, 50), true)
```

```
myPathFigure.AddSegment(myBezierSegment)
```

```
' Add the PathFigure to the PathGeometry
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

' VB .NET
Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

Dim myPath As new Path()
myPath.Data = myGeometryCollection

' Set the outline and the fill of the Path element.
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _
    MS Avalon.Windows.Media.Colors.Red)

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)
```

Geometry objects may also be rendered using the DrawingContext, which supplies a DrawGeometry method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

This example demonstrates how to draw shapes using the Geometry and Path elements in "Longhorn" markup language (code-named "XAML").

In the following example, a Path is drawn on a Canvas. Several Geometry elements are assigned to the Path element's Data attribute.

```
<Canvas ID="root"
    Background="White"
    xmlns="http://schemas.microsoft.com/2003/xaml">
```

```
<Path ID="myPath"
```

```

Fill="Blue"
Stroke="Black"
StrokeThickness="5">

<Path.Data>
  <GeometryCollection>

    <LineGeometry StartPoint="50,50" EndPoint="300,50"/>
    <EllipseGeometry Center="440, 100" RadiusX="40" RadiusY="75"/>
    <RectangleGeometry >
      <RectangleGeometry.Rect>
        <Rect X="400" Y="225" Width="100" Height="50"/>
      </RectangleGeometry.Rect>
    </RectangleGeometry>

    <PathGeometry>
      <PathGeometry.Figures>
        <PathFigureCollection>
          <PathFigure>
            <PathFigure.Segments>
              <PathSegmentCollection>
                <StartSegment Point="400,100"/>
                <BezierSegment Point1="400,100" Point2="400,200" Point3="200,300"/>
                <BezierSegment Point1="400,300" Point2="400,100" Point3="200,50"/>
                <BezierSegment Point1="0,100" Point2="0,200" Point3="200,300"/>
                <BezierSegment Point1="0,300" Point2="0,100" Point3="200,50"/>
              </PathSegmentCollection>
            </PathFigure.Segments>
          </PathFigure>
        </PathFigureCollection>
      </PathGeometry.Figures>
    </PathGeometry>

  </GeometryCollection>

</Path.Data>
</Path>

</Canvas>

```

in the previous example, the PathFigure object, one of the shapes drawn inside the Path element, contains a StartSegment but no CloseSegment; if a CloseSegment were added to the figure, a line would be drawn from the last segment in the collection back to the starting point of the figure.

LineSegment Class

Definition: Represents a line between two points. Unlike LineGeometry objects, LineSegment must be contained within a PathFigure.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited

	from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this LineSegment.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this LineSegment that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
LineSegment	Set to true if the line is stroked when a Pen is used to render the segment.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from PathSegment.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from

ReferenceEquals	Changeable. Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Point	Gets or sets the end point of the line segment.
PointAnimations	Gets or sets a collection of PointModifier objects that animate the line segment's end point.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Use a PathFigure object to store LineSegment objects and other segments. The LineSegment class does not contain a property for the line's start point. The line's starting point is the current point of the PathFigure object to which the line is added.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();
```

```
// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));
```

```
' VB .NET
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

```
' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))
```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));

myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));

myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));

MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);

myPathFigure.AddSegment(myBezierSegment);

myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))

Dim myBezierSegment As new BezierSegment(_
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(200, 50), true)

myPathFigure.AddSegment(myBezierSegment)

' Add the PathFigure to the PathGeometry
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```

// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

' VB .NET
Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

Dim myPath As new Path()
myPath.Data = myGeometryCollection

' Set the outline and the fill of the Path element.
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _
    MS Avalon.Windows.Media.Colors.Red)

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)

```

Geometry objects may also be rendered using the `DrawingContext`, which supplies a `DrawGeometry` method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

MatrixTransform Class

Definition: Creates an arbitrary affine matrix transformation used to manipulate objects or coordinate systems in a two-dimensional plane.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .
CloneCore	Implementation of <code>Animatable.CloneCore</code> .
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .

Copy	Creates a copy of this Transform. Inherited from Transform.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this MatrixTransform.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane. Inherited from Transform.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane. Inherited from Transform.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane. Inherited from Transform.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane. Inherited from Transform.
CreateTranslation	Creates a transformation used for translating in the x- and y-directions in a two-dimensional plane. Inherited from Transform.
DisableCore	Inherited from Transform.
Dispose	Dispose resources associated with transform. Inherited from Transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Transform.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Transform.
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base values set to the current animated values. Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this MatrixTransform that represents its current state.
GetHasAnimations	Inherited from Transform.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Transform.
GetIsOverridingBaseValue	Inherited from Transform.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Transform.
MatrixTransform	Initializes a new instance of the MatrixTransform class.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from

	Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimeline	Inherited from Transform.
SetDefaultParentTimelineCore	Inherited from Transform.
ToString	Not implemented. Use Object.ToString instead. Inherited from Transform.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the transformation has animations.
Identity	Identity transformation. Inherited from Transform.
IsAnimating	Returns true if the transformation contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently

modifiable. Inherited from Changeable.

IsOverridingBaseValue

Matrix

Gets or sets the Matrix structure that defines this transformation.

Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

StatusOfNextUse

UIContext

Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Value

Gets the current matrix transformation as a Matrix object.

Use the MatrixTransform class to create custom transformations not provided by the RotateTransform, SkewTransform, ScaleTransform, or TranslateTransform classes.

A 3x3 matrix is used for transformations in a two-dimensional x-y plane. Affine transformation matrices can be multiplied to form any number of linear transformations, such as rotation and skew (shear), followed by translation. An affine transformation matrix has its final column equal to (0, 0, 1), so only the members in the first two columns need to be specified.

An "Avalon" Matrix has the following structure:

M11 M12 0

M21 M22 0

OffsetX OffsetY 1

The members in the last row, OffsetX and OffsetY, represent translation values.

In methods and properties the transformation matrix is usually specified as a vector with only six members, as follows:

(M11, M12, M21, M22, OffsetX, OffsetY)

MediaData Class

Definition: MediaData. Use to playback Audio/Video content.

Method	Description
BeginIn	Schedules an interactive begin time.
CloneCore	Clones this MediaData.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable.
Dispose	Dispose the object.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.

EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null.
EndIn	Schedules an interactive end time.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Finalizes the MediaData.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MediaData	Creates a new MediaData.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
Pause	Pauses this media.
Play	Begins playback of media.
PropagateEventHandler	Propogates event handler to the timeline
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this media.
Seek	Moves the timeline for this media.
ToString	Persist MediaData in a string
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Accesses the Acceleration SMIL attribute.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.

AutoReverse	Accesses the AutoReverse SMIL attribute.
Begin	Accesses the Begin SMIL attribute.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	The current repetition iteration.
CurrentTime	The current time local to this media.
Deceleration	Accesses the Deceleration SMIL attribute.
Duration	Accesses the Duration SMIL attribute.
End	Accesses the End SMIL attribute.
EndSync	Accesses the EndSync SMIL attribute.
Fill	Accesses the Fill SMIL attribute.
FillDefault	Accesses the FillDefault SMIL attribute.
HasAudio	True if the media has audio output.
HasChanged	True if the media has changed since the last tick.
HasVideo	True if the media has a visual output.
Height	Get the video Height in pixels
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	True if the media is active, false otherwise.
IsEnabled	True if the media is enabled, false otherwise.
IsForwardProgressing	True if the media is moving from past to future.
IsOverridingBaseValue	True if the media is either changing or in a fill state, false otherwise.
IsPaused	True if this media is paused.
IsReversed	True if this media is in a reverse period.
MediaDuration	Returns the native media duration.
Mute	Accesses the mute state of media playback.
ParentTimeline	Accesses the ParentTimeline attribute.
Progress	The current progress of the media, from 0 to 1.
RepeatCount	Accesses the RepeatCount SMIL attribute.
RepeatDuration	Accesses the RepeatDuration SMIL attribute.
Restart	Accesses the Restart SMIL attribute.
RestartDefault	Accesses the RestartDefault SMIL attribute.
Speed	Accesses the Speed SMIL attribute.
State	Returns the current state of the media. Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
StatusOfNextUse	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Volume	Accesses the volume of media playback.
Width	Get the video Width in pixels

MediaSystem Class

Definition: The MediaSystem class controls the media layer.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MediaSystem	Initializes a new instance of the MediaSystem class.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Shutdown	This deinitializes the MediaSystem and frees any resources that it maintains.
StartFastAnimation	Calling this function causes the the MediaSystem to render as fast as possible.
Startup	This function initializes the MediaSystem. It must be called before any functions in the Media namespace can be used.
StopFastAnimation	This function will cease attempting to render as fast as possible. Rendering will still occur as normal.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
CompositionDevice	Returns a handle to the global composition device.
RenderDevice	Returns a handle to the global render device.

NineGridBrush Class

Definition: Fills an entire area with an image. Portions of the image are stretched to fit within defined margins.

Method	Description
CloneCore	Implementation of Animatable.CloneCore.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	Inherited from Brush.

EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Brush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
NineGridBrush	
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.

SetDefaultParentTimelineCore	Inherited from Brush.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
BottomBorder	BottomBorder - This accessor allows read only access to the BottomBorder property
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
GlyphImageSource	
HasAnimations	Gets a Boolean that indicates whether the brush has animations. Inherited from Brush.
ImageSource	
IsAnimating	Inherited from Brush.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Inherited from Brush.
LeftBorder	LeftBorder - This accessor allows read only access to the LeftBorder property
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.
RightBorder	RightBorder - This accessor allows read only access to the RightBorder property
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
TopBorder	TopBorder - This accessor allows read only access to the TopBorder property

Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

A NineGridBrush is very similar to an image brush in that it fills an area with a bitmap image. With a NineGridBrush, however, the image is divided into nine regions or grids by four borders. For more information, see the NineGridBrush page.

The following image is a typical bitmap that can be divided into nine grids.

The following image shows the same bitmap with nine grid borders highlighted with red lines.

The advantage of defining separate regions of the bitmap is that the image can be stretched without distorting the details around the edges or corners. If the image is stretched horizontally, only the center grid of each row is transformed, leaving the right and left edge unchanged. Similarly, if the image is stretched vertically, only the center grid of each column is transformed, leaving the top and bottom edges unchanged. The following image shows the previous NineGridBrush after it has been stretched in both directions.

The simplest way to fill a control with a NineGridBrush in "Longhorn" markup language (code-named "XAML") is to use the NineGrid keyword, followed by the file containing the image, and the offsets between the edges of the image and the borders. Offsets are always specified in the same order: left margin, right margin, top margin, bottom margin. The following image shows a set of borders with the margins labeled A, B, C, and D.

The syntax to create these borders for an image called myimage.png would look like this:
 NineGrid myimage.png A B C D

PathFigure Class

Definition: Represents a sub-section of a geometry, a single connected series of two-dimensional geometric segments.

Method	Description
AddSegment	Adds the specified segment to the figure.
ArcTo	Creates an ArcSegment, an elliptical arc, from the current point to the specified end point and adds it to the path. The path's current point is set to the end point of the arc.
BezierTo	Creates a BezierSegment, a cubic Bézier curve, and then adds it to the path. The BezierSegment is created from the current point to the specified end point using the two specified control points. The current point of the path is set to the end point of the curve.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Close	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is

Copy	Unchangeable. Inherited from Changeable.
DisableCore	Creates a copy of this PathFigure.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this PathFigure that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetFlattenedPathFigure	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
LineTo	Creates a line from the current point (the end point of the last segment defined in the current figure's Segments property) to the specified point.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PathFigure	
PolyBezierTo	Creates a series of cubic Bézier curves and adds them to the path. The path's current point is set to the destination point of the last curve.
PolyLineTo	Creates a series of lines and adds them to the path. The path's current point is set to the destination point of the last line.
PolyQuadraticBezierTo	Creates a series of quadratic Bézier curves and adds them to the path. The path's current point is set to the destination point of the

PropagateEventHandler	last curve.
QuadraticBezierTo	Creates a quadratic Bézier curve and adds it to the path. The curve is created from the current point to the specified destination point using the specified control point. The path's current point is set to the destination point of the curve.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
StartAt	Specifies the beginning point of the figure. A figure must have one, and only one, beginning point.
StrokeNewSegments	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether any PathSegment that belongs to the figure has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFilled	
IsOverridingBaseValue	

Segments

StatusOfNextUse

Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

UIContext

Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Each PathGeometry object defines a collection of PathFigure objects. Each of the PathFigure objects is composed of one or more PathSegment objects, such as ArcSegment and BezierSegment, which actually define their shape.

PathFigureCollection Class

Method	Description
Add	Adds a PathFigure to the end of the collection.
AddRange	
Clear	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	
Copy	
CopyTo	Copies the entire PathFigureCollection to a compatible one-dimensional Array, starting at the specified index of the target array.
DisableCore	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionReader	
EmbeddedAnimationCollectionWriter	
EmbeddedChangeableReader	
EmbeddedChangeableWriter	
EnableCore	Determines whether two Object instances are equal. Inherited from Object.
Equals	
Finalize	
GetCurrentValue	Returns a non-animated version of this PathFigureCollection that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that

GetEnumerator	represents its current state. Inherited from Animatable.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PathFigureCollection	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
SetRange	

ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity Count	
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize IsOverridingBaseValue Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PathFigureConverter Class

Definition: Path Figure Converter

Method	Description
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	TypeConverter method override.

ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	TypeConverter method implementation.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PathFigureConverter	
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

PathGeometry Class

Definition: Represents a complex shape that may be composed of arcs, curves, ellipses, lines, and rectangles.

Method	Description
AddFigure	Adds a PathFigure, a collection of PathSegment objects that describe a shape, to the path.
AddGeometry	Converts the specified Geometry into a collection of PathFigure objects and adds it to the path. If the specified Geometry is animated, the conversion from Geometry to PathFigure may result in some loss of information.
AddPointAndTypes	
CloneCore	Returns a modifiable shallow or deep clone of the current object.

	This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathGeometry.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this Geometry. Inherited from Geometry.
DisableCore	
Dispose	Dispose resources associated with geometry.
DoesContain	Returns if point is inside the geometry. Inherited from Geometry.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Geometry.
GetBounds	Returns the bounds of a Geometry, widened according to the characteristics of the specified pen. Inherited from Geometry.
GetCurrentValue	Returns a non-animated version of this PathGeometry that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this Geometry that represents its current state. Inherited from Geometry.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetOutlinedPathGeometry	
GetType	Gets the Type of the current instance. Inherited from Object.
GetWidenedPathGeometry	
IAddChild.AddChild	
IAddChild.AddText	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.

OnChanged	Called when the current object is modified. Classes that derive from Changeable should call this method after they have been modified. Inherited from Changeable.
PathGeometry PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Implementation of Changeable.ValidateObjectState.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
Bounds	Gets a Rect that specifies the bounding box of a Geometry. This method does not take any pens into account. Inherited from Geometry.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
Figures	
FillRule	
HasAnimations	
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a

	Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets the Transform object applied to a Geometry. Inherited from Geometry.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Each PathGeometry object defines a collection of PathFigure objects. Each of the PathFigure objects is composed of one or more PathSegment objects, such as ArcSegment and LineSegment, which actually define their shape.

The filled area of the PathGeometry is defined by taking all of the contained PathFigure objects that have their IsFilled property set to true and applying the FillRule to determine the enclosed area.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's

Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();
```

```
// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));
```

```
' VB .NET
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

```
' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))
```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));

myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));

myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));

MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);

myPathFigure.AddSegment(myBezierSegment);

myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))

Dim myBezierSegment As new BezierSegment( _
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
```



```
new MS Avalon.Windows.Point(200, 50), true)
```

```
myPathFigure.AddSegment(myBezierSegment)
```

```
' Add the PathFigure to the PathGeometry  
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
```

```
GeometryCollection myGeometryCollection = new GeometryCollection();  
myGeometryCollection.Add(myLineGeometry);  
myGeometryCollection.Add(myEllipseGeometry);  
myGeometryCollection.Add(myRectangleGeometry);  
myGeometryCollection.Add(myPathGeometry);
```

```
Path myPath = new Path();  
myPath.Data = myGeometryCollection;
```

```
// Set the outline and the fill of the Path element.  
myPath.Stroke = Brushes.Blue;  
myPath.StrokeThickness = new Length(5);  
SolidColorBrush solidFill = new SolidColorBrush();  
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);
```

```
// Add the Path element to a Canvas.  
myCanvas.Children.Add(myPath);
```

```
' VB .NET
```

```
Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()  
myGeometryCollection.Add(myLineGeometry)  
myGeometryCollection.Add(myEllipseGeometry)  
myGeometryCollection.Add(myRectangleGeometry)  
myGeometryCollection.Add(myPathGeometry)
```

```
Dim myPath As new Path()  
myPath.Data = myGeometryCollection
```

```
' Set the outline and the fill of the Path element.  
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue  
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)  
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()  
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _  
    MS Avalon.Windows.Media.Colors.Red)
```

```
' Add the Path element to a Canvas.  
myCanvas.Children.Add(myPath)
```

Geometry objects may also be rendered using the DrawingContext, which supplies a DrawGeometry method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

PathGeometryConverter Class
Definition: PathGeometryConverter

Method	Description
CanConvertFrom	CanConvertFrom
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	TypeConverter method override.
CanConvertTo	Inherited from TypeConverter.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	ConvertFrom
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertFromString	ConvertFromString
ConvertTo	Inherited from TypeConverter.
ConvertTo	TypeConverter method implementation.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PathGeometryConverter	
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

PathSegment Class

Definition: An abstract class that represents a segment of a PathFigure object. Classes that derive from PathSegment, such as ArcSegment, BezierSegment, and LineSegment, represent specific types of geometric segments.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Subclasses must implement this to provide clones of themselves. Inherited from Animatable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.

PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Returns true if any of the properties on this Changeable have animations attached. Inherited from Animatable.
IsAnimating	Returns true if any of the properties on this Changeable have animations attached that are currently animating their value. Inherited from Animatable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Returns true if any of the properties on this Changeable have animations attached that are currently affecting their value either by animating it or holding it in a fill state. Inherited from Animatable.
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a

Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

UIContext Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

This example demonstrates how to draw shapes using the Geometry, PathFigure, and PathSegment classes. In this example, several shapes are drawn using Geometry objects and are displayed using a Path element.

There is a Geometry class for each basic geometric shape: LineGeometry, EllipseGeometry, and RectangleGeometry. Complex shapes, such as polygons and shapes with curved segments, may be created using a PathGeometry.

In the following example, a LineGeometry, EllipseGeometry, and a RectangleGeometry object are used to create a line, an ellipse, and a rectangle.

```
// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))
```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```
// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
```

```
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();
```

```
// PathFigure objects must have a defined start point before  
// other segments can be added.
```

```
myPathFigure.StartAt(new Point(200,50));
```

```
' VB .NET
```

```
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
```

```
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()
```

```
' PathFigure objects must have a defined start point before
```

```
' other segments can be added.
```

```
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))
```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
```

```
myPathFigure.BezierTo(  
    new Point(400, 100), new Point(400, 200), new Point(200, 300));
```

```
myPathFigure.BezierTo(  
    new Point(400, 300), new Point(400, 100), new Point(200, 50));
```

```
myPathFigure.BezierTo(  
    new Point(0, 100), new Point(0, 200), new Point(200,300));
```

```
MSAvalon.Windows.Media.BezierSegment myBezierSegment =  
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);
```

```
myPathFigure.AddSegment(myBezierSegment);
```

```
myPathGeometry.Figures.Add(myPathFigure);
```

```
' VB .NET
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _  
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _  
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))
```

```
myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _  
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))
```

```
Dim myBezierSegment As new BezierSegment(_  
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _  
    new MSAvalon.Windows.Point(200, 50), true)
```

```
myPathFigure.AddSegment(myBezierSegment)
```

```
' Add the PathFigure to the PathGeometry
```

```
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);

// Add the Path element to a Canvas.
myCanvas.Children.Add(myPath);

' VB .NET
Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()
myGeometryCollection.Add(myLineGeometry)
myGeometryCollection.Add(myEllipseGeometry)
myGeometryCollection.Add(myRectangleGeometry)
myGeometryCollection.Add(myPathGeometry)

Dim myPath As new Path()
myPath.Data = myGeometryCollection

' Set the outline and the fill of the Path element.
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _
    MS Avalon.Windows.Media.Colors.Red)

' Add the Path element to a Canvas.
myCanvas.Children.Add(myPath)
```

Geometry objects may also be rendered using the DrawingContext, which supplies a DrawGeometry method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

PathSegmentCollection Class

Definition: Represents a list of PathSegment objects.

Method	Description
Add	Adds a PathSegment to a list and returns the index at which the segment is added.
AddRange	Adds the segments contained in the specified PathSegmentCollection to the collection.

Clear	Clears the collection of all segments and resets Count to zero.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a Boolean that indicates whether the specified PathSegment is contained within the collection.
Copy	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	Copies the entire PathSegmentCollection to a compatible one-dimensional Array, starting at the specified index of the target array.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this PathSegmentCollection that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	

LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PathSegmentCollection	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.

Capacity	
Count	
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
IsOverridingBaseValue	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PathSegmentConverter Class

Definition: PathSegmentConverter

Method	Description
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	TypeConverter method override.
ConvertFrom	Inherited from TypeConverter.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	TypeConverter method implementation.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in

	hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PathSegmentConverter	
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

Pen Class

Definition: Describes how a shape is outlined.

Method	Description
CloneCore	Implementation of Animatable.CloneCore.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this Pen.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	
GetCurrentValue	Returns a non-animated version of this Pen that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.

GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
Pen	Initializes a new instance of the Pen class.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
Brush	Gets or sets the fill of an outline.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DashArray	Gets or sets the pattern of dashes and gaps used to stroke paths.
DashCap	Gets or sets a PenDashCap enumeration that specifies how the ends of each dash is drawn.

DashOffset	Gets or sets the offset to the start location of the dash pattern. When you set the value you specify that the dash pattern should start at a location other than the origin of the stroke.
DashOffsetAnimations	Gets or sets a collection of DoubleModifier objects that animate a pen's DashOffset property.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
EndLineCap	Gets or sets the type of shape to use at the end of a stroke.
HasAnimations	Gets a Boolean that indicates whether the pen or its Brush has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
LineJoin	Gets or sets the type of joint used at the vertices of a shape's outline.
MiterLimit	Gets or sets the limit on the ratio of the miter length to the Thickness of a Pen.
StartLineCap	Gets or sets the type of shape to use at the beginning of a stroke.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Thickness	Gets or sets the width of the stroke.
ThicknessAnimations	Gets or sets a collection of DoubleModifier objects that animate the pen's Thickness.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PixelFormats Class

Definition: PixelFormats - The collection of supported Pixel Formats.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PixelFormats	Determines whether the specified Object instances are the same instance. Inherited from Object.

ReferenceEquals

ToString

Returns a String that represents the current Object. Inherited from Object.

Property	Description
ABGR128	ABGR128: 128 bpp extended format; Gamma is 1.0
ARGB32	ARGB32: 32 bpp SRGB format
ARGB64	ARGB64: 64 bpp extended format; Gamma is 1.0
BGR24	BGR24: 24 bpp SRGB format
BlackWhite	BlackWhite: Monochrome, 2-color image, black and white only.
CMYK32	CMYK32: 32 bpp format
DontCare	DontCare: for situations when the pixel format may not be important
Gray1	Gray1: Image with 2 shades of gray (same as BlackWhite)
Gray2	Gray2: Image with 4 shades of gray
Gray32	Gray32: 32 bpp Gray-scale format; Gamma is 1.0
Gray4	Gray4: Image with 16 shades of gray
Gray8	Gray8: Image with 256 shades of gray
Indexed1	Indexed1: Paletted image with 2 colors.
Indexed2	Indexed2: Paletted image with 4 colors.
Indexed4	Indexed4: Paletted image with 16 colors.
Indexed8	Indexed8: Paletted image with 256 colors.
PABGR128	PABGR128: 128 bpp extended format; Gamma is 1.0
PARGB32	PARGB32: 32 bpp SRGB format
PARGB64	PARGB64: 64 bpp extended format; Gamma is 1.0
RGB24	RGB24: 24 bpp SRGB format
RGB32	RGB32: 32 bpp SRGB format
RGB48	RGB48: 48 bpp extended format; Gamma is 1.0
RGB555	RGB555: 16 bpp SRGB format
RGB565	RGB565: 16 bpp SRGB format
Undefined	Undefined: the pixel format is not known

PointCollection Class

Method	Description
Add	
AddRange	
Clear	
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	

CopyTo	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableReader	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
EmbeddedChangeableWriter	Determines whether two Object instances are equal. Inherited from Object.
Equals	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
Finalize	
GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PointCollection	
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.

ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	
Count	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PointCollectionConverter Class

Definition: PointCollectionConverter - Converter class for converting instances of other types to and from PointCollection instances.

Method	Description
CanConvertFrom	CanConvertFrom - Returns whether or not this class can convert from a given type.
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.

CanConvertTo	CanConvertTo - Returns whether or not this class can convert to a given type.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	ConvertFrom - Attempt to convert to a PointCollection from the given object
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	ConvertTo - Attempt to convert a PointCollection to the given type
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PointCollectionConverter	
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.

PointHitTestParameters Class

Definition: This is the class for specifying parameters hit testing with a point.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.

GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PointHitTestParameters	The constructor takes the point to hit test with.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
HitPoint	The point to hit test against.

PointHitTestResult Class

Definition: This class returns the point and visual hit during a hit test pass.

Method	Description
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
PointHitTestResult	This constructor takes a visual and point representing a hit.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Point	The point in local space of the hit visual.
Visual	Returns the visual that was hit. Inherited from HitTestResult.

PolyBezierSegment Class

Definition: PolyBezierSegment

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.

Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
Copy	Creates a copy of this <code>PolyBezierSegment</code> .
DisableCore	Inherited from <code>PathSegment</code> .
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from <code>Animatable</code> .
EmbeddedAnimationCollectionWriter	Inherited from <code>Animatable</code> .
EmbeddedChangeableReader	Inherited from <code>Animatable</code> .
EmbeddedChangeableWriter	Inherited from <code>Animatable</code> .
EnableCore	Inherited from <code>PathSegment</code> .
Equals	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
Finalize	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
GetCurrentValue	Returns a non-animated version of this <code>GradientStop</code> that represents its current state. Inherited from <code>PathSegment</code> .
GetCurrentValue	Returns a non-animated version of this <code>PolyBezierSegment</code> that represents its current state.
GetCurrentValue	Returns a non-animated version of this <code>Animatable</code> that represents its current state. Inherited from <code>Animatable</code> .
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
GetType	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
MakeUnchangeable	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
MakeUnchangeableCore	Implementation of <code>MakeUnchangeableCore</code> .
MemberwiseClone	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
ModifyHandlerIfChangeable	Adds or removes a <code>Changed</code> event handler to or from the specified <code>Changeable</code> object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
OnChanged	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
PolyBezierSegment	
PropagateEventHandler	Implementation of <code>PropagateEventHandler</code> .
ReadPreamble	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
ReferenceEquals	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this <code>Changeable</code> which are inheriting their time

SetDefaultParentTimelineCore	parent. Inherited from Animatable. Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Points	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PolyLineSegment Class Definition: PolyLineSegment

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.

CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this PolyLineSegment.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this PolyLineSegment that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PolyLineSegment	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from

	Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Points	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PolyQuadraticBezierSegment Class
Definition: PolyQuadraticBezierSegment

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this PolyQuadraticBezierSegment.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this PolyQuadraticBezierSegment that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive

	from Changed should call this method after they have been modified. Inherited from Changeable.
PolyQuadraticBezierSegment	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Points	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a

	complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

PrintContext Class

Definition: PrintContext holds state and context for a printer interaction

Method	Description
AddPage	Add a visual as the last page
Dispose	Dispose cleans up the state associated
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Finalizer for ImageData
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
Print	Print current job
PrintContext	Connect to default printer
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SelectPrintQueue	SelectPrintQueue
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
JobTicket	JobTicket property
Output	Output property

This example demonstrates how to use a PrintContext object to write to a printer. The following code demonstrates how to print a Visual (the VectorPage in this example) to the default printer. The VectorPage in this example derives from DrawingVisual and is used to draw several shapes. Two **VectorPage** objects are printed by calling the AddPage method twice.

[C#]

```
public static void PrintPaginatedVisualToDefaultPrinter() {
    // create a print context for my default printer
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext();

    // Add two pages to the default rendition
    pc.AddPage(new VectorPage());
    pc.AddPage(new VectorPage());

    //Print
    pc.Print();
}
```

The next example demonstrates how to print a Visual to a printer that the user selects. The `SelectPrintQueue` method is used to produce a dialog box that enables the user to select the printer, number of copies to print, and other options.

[C#]

```
public static void PrintPaginatedVisualToSelectedPrinter() {
    // Let user select a printer queue using common dialog box
    System.Printing.PrintSubSystem.PrintQueue queue =
        MSAvalon.Windows.Media.PrintContext.SelectPrintQueue();

    if (queue != null) {
        // Create MSAvalon.Windows.Media.PrintContext if OK button was clicked
        MSAvalon.Windows.Media.PrintContext pc =
            new MSAvalon.Windows.Media.PrintContext(queue, "Printing Example");

        // Add a page of visual
        pc.AddPage(new VectorPage());

        // Print
        pc.Print();
    }
}
```

The next example demonstrates how to print to landscape and portrait page orientations. In the following code, a `JobTicket` is obtained and used to set the page orientation. The `DimensionPage` in this example is a `DrawingVisual` that adjust the size of its drawing to the size of the output page. The `GetPaperWidthHeight` method in this example is used to retrieve the page size information from the `JobTicket`.

[C#]

```
public static void MixedOrientationPrint() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "Landscape Example");

    Microsoft.Printing.JobTicket.JobTicket jt = pc.JobTicket;

    // Print a page with landscape orientation.
    {
        jt.PageOrientation.Value =
            System.Printing.Configuration.PrintSchema.OrientationValues.Landscape;

        double width, height;

        GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

        // Add a page of visual
        pc.AddPage(new DimensionPage(height, width));
    }
}
```

```

// Print a page with portrait orientation.
{
    jt.PageOrientation.Value =
        System.Printing.Configuration.PrintSchema.OrientationValues.Portrait;

    double width, height;

    GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

    // Add a page of visual
    pc.AddPage(new DimensionPage(width, height));
}

// Print
pc.Print();
}

internal static void GetPaperWidthHeight(Microsoft.Printing.JobTicket.JobTicket jt,
    System.Printing.PrintSubSystem.PrintQueue pq, out double width, out double height) {

    Microsoft.Printing.DeviceCapabilities.DeviceCapabilities devcap =
        new Microsoft.Printing.DeviceCapabilities.DeviceCapabilities(
            pq.AcquireDeviceCapabilities(null));

    devcap.LengthUnitType = System.Printing.Configuration.PrintSchema.LengthUnitTypes.Inch;

    width = (double) devcap.PageCanvasSizeCap.CanvasSizeX * 96;
    height = (double) devcap.PageCanvasSizeCap.CanvasSizeY * 96;
}

```

The next example demonstrates how to print based on the `GetPage` event. The `GeneratePage` method is used to handle the event.

[C#]

```

public static void EventDriven() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    // Create MSAvalon.Windows.Media.PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "EventDriven(2 pages)");

    pc.GetPage += new MSAvalon.Windows.Media.PrintContext.GetPageEventHandler(GetAPage);

    // Print
    pc.Print();
}

private static MSAvalon.Windows.Media.Visual GeneratePage(object sender,
    int pageNo, MSAvalon.Windows.Media.GetPageEventArgs ev) {

    if (pageNo == 0) {
        return new VectorPage();
    }
}

```

```

    }
    else if (pageNo == 1) {
        return new ImagePage();
    }
    else return null;
}

```

When running the sample and performing this operation, a dialog box with an assertion error message may appear. Click the Ignore All button to ignore the assertion and print the pages.

The final example shows how to print to a file by using the PrintContext object's Output property.

[C#]

```

public static void PrintToFile() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    // Create PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "PrintToFile");

    pc.Output = "test.prn";

    // Add a page of visual
    pc.AddPage(new VectorPage());

    // Print
    pc.Print();
}

```

This example demonstrates how to use a PrintContext object to print a rendered "Longhorn" markup language (code-named "XAML") file.

In the following example, the LoadContent method is used to load, parse, and return a "XAML" file's root UIElement. The UIElement, page, is then configured with a LayoutManager, added to the PrintContext object using the AddPage method, and printed with the Print method.

[C#]

```

public static void PrintXamlFile(string xamlFilePath) {
    // Let user select a printer queue using common dialog box
    System.Printing.PrintSubSystem.PrintQueue queue =
        MSAvalon.Windows.Media.PrintContext.SelectPrintQueue();

    if (queue != null) {

        MSAvalon.Windows.Media.PrintContext pc =
            new MSAvalon.Windows.Media.PrintContext(queue, "Element printing test");

        MSAvalon.Windows.UIElement page = LoadContent(xamlFilePath);
        MSAvalon.Windows.LayoutManager lm = new MSAvalon.Windows.LayoutManager(page);

        double width, height;

        GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);
    }
}

```

```

        lm.Size = new MSAvalon.Windows.Size(width, height);
        lm.UpdateLayout();

        pc.AddPage(page);
        pc.Print();
    }
}

```

The LoadContent method used in the previous example is used to load the "XAML" file, parse it, and return its root UIElement. The GetPaperWidthHeight method in this example is used to retrieve the page size information from the JobTicket.

[C#]

```

public static MSAvalon.Windows.UIElement LoadContent(string xamlFilePath) {

    System.IO.Stream xamlFileStream = System.IO.File.OpenRead(xamlFilePath);

    MSAvalon.Windows.UIElement root = null;

    try {
        MSAvalon.Windows.Serialization.ParserContext pc =
            new MSAvalon.Windows.Serialization.ParserContext();

        System.Security.PermissionSet ps =
            MSAvalon.Windows.TrustManagement.TrustManager.GetDefaultPermissions();

        System.Security.Permissions.FileIOPermission fiop =
            new System.Security.Permissions.FileIOPermission(
                System.Security.Permissions.FileIOPermissionAccess.AllAccess,
                System.IO.Path.GetFullPath(".\\"));

        ps.AddPermission(fiop);

        root =
            (MSAvalon.Windows.UIElement) MSAvalon.Windows.Serialization.Parser.LoadXml(
                xamlFileStream, null, pc, null, ps);
    }
    catch (Exception e) {
        Console.WriteLine("Load Failed:");
        Console.WriteLine(e);
    }
    finally {
        // done with the stream
        xamlFileStream.Close();
    }

    return root;
}

```

```

internal static void GetPaperWidthHeight(Microsoft.Printing.JobTicket.JobTicket jt,
    System.Printing.PrintSubSystem.PrintQueue pq, out double width, out double height) {

    Microsoft.Printing.DeviceCapabilities.DeviceCapabilities devcap =
        new Microsoft.Printing.DeviceCapabilities(

```

```

        pq.AcquireDeviceCapabilities(null));

    devcap.LengthUnitType = System.Printing.Configuration.PrintSchema.LengthUnitTypes.Inch;

    width = (double) devcap.PageCanvasSizeCap.CanvasSizeX * 96;
    height = (double) devcap.PageCanvasSizeCap.CanvasSizeY * 96;
}

```

QuadraticBezierSegment Class

Definition: QuadraticBezierSegment

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from PathSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this QuadraticBezierSegment.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this QuadraticBezierSegment that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from

MemberwiseClone	PathSegment. Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler QuadraticBezierSegment	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently

IsOverridingBaseValue	modifiable. Inherited from Changeable.
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Point1	
Point1Animations	
Point2	
Point2Animations	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

RadialGradientBrush Class

Definition: Defines a radial gradient used to fill an object. A focal point defines the beginning of the gradient, and a circle defines the end point of the gradient.

Method	Description
AddStop	Adds a gradient stop to the brush. Inherited from GradientBrush.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from GradientBrush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	Inherited from GradientBrush.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.

EnableCore	Inherited from GradientBrush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this GradientBrush that represents its current state. Inherited from GradientBrush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time. That is, the returned brush is a snapshot of the current object at the point in time at which this method was called.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Inherited from GradientBrush.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
RadialGradientBrush	Initializes a new instance of the RadialGradientBrush class.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time

SetDefaultParentTimelineCore	parent. Inherited from Animatable.
ToString	Inherited from GradientBrush.
ValidateObjectState	Returns a String that represents the current Object. Inherited from Object.
WritePostscript	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePreamble	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Center	Gets or sets the center of the outermost circle of the radial gradient.
CenterAnimations	Gets or sets a collection of PointModifier objects that animate the brush's Center property.
ColorInterpolationMode	ColorInterpolationMode - Read only accessor of the ColorInterpolationMode property. This property controls how the colors in Gradient are interpolated. Default is ColorInterpolationMode.PerceptuallyLinearGamma. Inherited from GradientBrush.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
Focus	Gets or sets the location of the two-dimensional focal point that defines the beginning of the gradient.
FocusAnimations	Gets or sets a collection of PointModifier objects that animate the brush's Focus property.
GradientStops	Gets or sets the gradient stops (transition points) of a brush. Inherited from GradientBrush.
HasAnimations	Gets a Boolean that indicates whether the brush has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
MappingMode	Gets or sets a BrushMappingMode enumeration that specifies whether the gradient brush's positioning coordinates are absolute or relative to the output area. Inherited from GradientBrush.
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the

	brush. Inherited from Brush.
RadiusX	Gets or sets the horizontal radius of the outermost circle of the radial gradient.
RadiusXAnimations	Gets or sets a collection of DoubleModifier objects that animate the brush's RadiusX property.
RadiusY	Gets or sets the vertical radius of the outermost circle of a radial gradient.
RadiusYAnimations	Gets or sets a collection of DoubleModifier objects that animate the brush's RadiusY property.
SpreadMethod	Gets or sets the type of spread method that specifies how to draw a gradient that starts or ends inside the bounds of the object to be painted. Inherited from GradientBrush.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The RadialGradientBrush is similar in programming model to the LinearGradientBrush. However, the linear gradient has a start and end point to define the gradient vector, while the radial gradient has a circle along with a focal point to define the gradient behavior. The circle defines the end point of the gradient. In other words, a gradient stop at 1.0 defines the color at the circle's circumference. The focal point defines the center of the gradient. A gradient stop at 0.0 defines the color at the focal point.

The following image shows a rectangle filled with a radial gradient. The radial gradient that goes from white to grey. The outside circle represents the gradient circle while the red dot denotes the focal point. This gradient has its SpreadMethod set to Pad.

This example demonstrates how to use the Transform property of brushes to apply transformations to LinearGradientBrush and RadialGradientBrush fills. A LinearGradientBrush is used to fill the first two Rectangle elements. The difference between the rectangles is that the LinearGradientBrush in the second rectangle is rotated 45 degrees. The second pair of rectangles illustrates the before and after effect of a ScaleTransform by reducing a RadialGradientBrush to half its normal height.

```

    <Border xmlns="http://schemas.microsoft.com/2003/xaml"
    Background="#CCCCCC">

    <Canvas Height="40">

    <!-- Rectangle #1 is filled with a LinearGradientBrush. The gradient colors
    flow from left to right by default. -->

    <Rectangle RectangleLeft="10" RectangleTop="10"
    RectangleWidth="300" RectangleHeight="200">

    <Rectangle.Fill>
    <LinearGradientBrush>

```

```

    <LinearGradientBrush.GradientStops>
      <GradientStopCollection>
        <GradientStop Color="red" Offset="0"/>
        <GradientStop Color="yellow" Offset="1" />
        <GradientStop Color="blue" Offset="0.5"/>
        <GradientStop Color="white" Offset="0.2"/>
      </GradientStopCollection>
    </LinearGradientBrush.GradientStops>

  </LinearGradientBrush>
</Rectangle.Fill>

</Rectangle>

```

<!-- Rectangle #2 is identical to the first rectangle except that the Transform property rotates the LinearGradientBrush so that the gradient colors are rotated by 45 degrees. -->

```

<Rectangle RectangleLeft="320" RectangleTop="10"
  RectangleWidth="300" RectangleHeight="200">

  <Rectangle.Fill>
    <LinearGradientBrush>

      <LinearGradientBrush.Transform>
        <RotateTransform Angle="45" /> <!-- Rotation angle. -->
      </LinearGradientBrush.Transform>

      <LinearGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="red" Offset="0"/>
          <GradientStop Color="yellow" Offset="1" />
          <GradientStop Color="blue" Offset="0.5"/>
          <GradientStop Color="white" Offset="0.2"/>
        </GradientStopCollection>
      </LinearGradientBrush.GradientStops>

    </LinearGradientBrush>
  </Rectangle.Fill>

</Rectangle>

```

<!-- Rectangle #3 is filled with a RadialGradientBrush. -->

```

<Rectangle RectangleLeft="10" RectangleTop="250"
  RectangleWidth="300" RectangleHeight="200">

  <Rectangle.Fill>
    <RadialGradientBrush Focus="0.5,0.5">

      <RadialGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="red" Offset="0"/>

```

```

        <GradientStop Color="yellow" Offset="1"/>
        <GradientStop Color="blue" Offset="0.5"/>
    </GradientStopCollection>
</RadialGradientBrush.GradientStops>

</RadialGradientBrush>
</Rectangle.Fill>

</Rectangle>

```

<!-- Rectangle #4 is identical to the third rectangle except that the Transform property applies a ScaleTransform to the RadialGradientBrush so that the gradient is half its previous height. -->

```

<Rectangle RectangleLeft="320" RectangleTop="250"
    RectangleWidth="300" RectangleHeight="200">

    <Rectangle.Fill>

        <RadialGradientBrush Focus="0.5,0.5">
            <RadialGradientBrush.Transform>
                <ScaleTransform ScaleX="1" ScaleY="0.5" /><!-- Scale transform. -->
            </RadialGradientBrush.Transform>

            <RadialGradientBrush.GradientStops>
                <GradientStopCollection>
                    <GradientStop Color="red" Offset="0"/>
                    <GradientStop Color="yellow" Offset="1"/>
                    <GradientStop Color="blue" Offset="0.5"/>
                </GradientStopCollection>
            </RadialGradientBrush.GradientStops>

        </RadialGradientBrush>
    </Rectangle.Fill>

</Rectangle>

</Canvas>
</Border>

```

This example creates simple vertical, horizontal, and radial gradients and uses them to fill an element using "Longhorn" markup language (code-named "XAML").

In the following example, vertical and horizontal gradients are used to set the Fill property of two Rectangle elements. In this particular example, the gradients are described using simple notation: GradientType StartColor EndColor, where GradientType is VerticalGradient, HorizontalGradient, or RadialGradient. StartColor and EndColor can be predefined color names (such as Blue) or hexadecimal values.

```

<Canvas xmlns="http://schemas.microsoft.com/2003/xaml">

    <Rectangle
        Fill="VerticalGradient Blue Green"
        RectangleLeft="20"
        RectangleTop="20"

```

```

    RectangleWidth="100"
    RectangleHeight="100">
</Rectangle>

```

```

<Rectangle
    Fill="HorizontalGradient Blue Red"
    RectangleLeft="120"
    RectangleTop="120"
    RectangleWidth="100"
    RectangleHeight="100">
</Rectangle>

```

A vertical gradient is a linear gradient whose start and endpoints form a vertical line; likewise, a horizontal gradient is a linear gradient whose start and endpoints form a horizontal line. You can explicitly describe your own linear gradients using the following syntax:

LinearGradient StartPoint EndPoint StartColor EndColor, where StartPoint and EndPoint are the starting and ending coordinates, with each coordinate expressed as a pair of x and y values from 0 to 1, such as 0.1,0.1 and 0.5,0.5. These values indicate the relative position of the start or end point. An endpoint of 0.5,0.5 would be located 50 percent to the right of the fill area and 50 percent of the way from the top of the area—the middle of the shape.

In the following example, the Fill property of a Rectangle element is set by explicitly using a linear gradient.

```

<Rectangle
    Fill="LinearGradient 0.1,0.1 0.5,0.5 Blue Green"
    RectangleLeft="220"
    RectangleTop="220"
    RectangleWidth="100"
    RectangleHeight="100">
</Rectangle>

```

In the final example, the Fill property of a Rectangle element is set using a radial gradient.

```

<Rectangle
    Fill="RadialGradient Blue Red"
    RectangleLeft="320"
    RectangleTop="320"
    RectangleWidth="100"
    RectangleHeight="100">
</Rectangle>

```

```

</Canvas>

```

See [Create a Gradient with More Than Two Colors](#) for an example of how to create a gradient with more than two gradient stops.

To create vertical and horizontal gradients in code or using compound notation, use the LinearGradientBrush class and set its StartPoint and EndPoint properties so that they describe a vertical or horizontal line. To create radial gradients in code or using compound notation, use the RadialGradientBrush class.

This example demonstrates how to create a gradient that has more than two colors in "XAML". To create a gradient with more than two colors, add a GradientStopCollection to the gradient's GradientStops property. Next, add GradientStop objects to the GradientStopCollection, one for each color the gradient should contain. Set the Color and the Offset, a value from 0 to 1 that determines the relative position of the stop in the gradient, of each of the stops. The following example shows a Button whose Background is filled with a horizontal gradient that has four colors.

```

<Canvas ID="root"

```

```
xmlns="http://schemas.microsoft.com/2003/xaml">
```

```
<Button
  Canvas.Top="50"
  Canvas.Left="50"
  BorderBrush="Black"
  Width="200"
  Height="30">
  <Button.Background>
    <LinearGradientBrush >
      <LinearGradientBrush.GradientStops>
        <GradientStopCollection>
          <GradientStop Color="Red" Offset="0" />
          <GradientStop Color="Blue" Offset="0.25"/>
          <GradientStop Color="Orange" Offset="0.75"/>
          <GradientStop Color="Yellow" Offset="1"/>
        </GradientStopCollection>
      </LinearGradientBrush.GradientStops>
    </LinearGradientBrush>
  </Button.Background>
</Button>

</Canvas>
```

RectangleGeometry Class

Definition: Represents the geometry of a rectangle.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this Geometry. Inherited from Geometry.
Copy	Creates a copy of this RectangleGeometry.
DisableCore	Inherited from Geometry.
Dispose	Releases the resources associated with the object. Inherited from Geometry.
DoesContain	Returns if point is inside the geometry. Inherited from Geometry.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Geometry.
Equals	Determines whether two Object instances are equal. Inherited from Object.

Finalize	Inherited from Geometry.
GetBounds	Returns the bounds of a Geometry, widened according to the characteristics of the specified pen. Inherited from Geometry.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this Geometry that represents its current state. Inherited from Geometry.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Inherited from Geometry.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
RectangleGeometry	Initializes a new instance of the RectangleGeometry class.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from Geometry.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in

	as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
Bounds	Gets a <code>Rect</code> that specifies the bounding box of a <code>RectangleGeometry</code> . This method does not take any pens into account.
CanMakeUnchangeable	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
DefaultParentTimeline	The current parent <code>Timeline</code> associated with this <code>Animatable</code> . This will be the <code>Timeline</code> set to the <code>ParentTimeline</code> property of this <code>Animatable</code> if one has been set and if not, the <code>Timeline</code> last passed into the <code>SetDefaultParentTimeline</code> method. Inherited from <code>Animatable</code> .
HasAnimations	Gets a <code>Boolean</code> that indicates whether the geometry has animations.
IsAnimating	Returns true if the geometry contains active animations.
IsChangeable	Gets a <code>Boolean</code> that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
IsOverridingBaseValue	
RadiusX	Gets or sets the radius of the rounded corner where it connects with the upper and lower edges of the rectangle.
RadiusXAnimations	Gets or sets a collection of <code>DoubleModifier</code> objects that animate the geometry's <code>RadiusX</code> property.
RadiusY	Gets or sets the radius of the rounded corner where it connects with the left and right edges of the rectangle.
RadiusYAnimations	Gets or sets a collection of <code>DoubleModifier</code> objects that animate the geometry's <code>RadiusY</code> property.
Rect	Gets or sets the dimensions of the rectangle.
RectAnimations	Gets or sets a collection of <code>RectModifier</code> objects that animate the rectangle's dimensions.
StatusOfNextUse	Gets or sets a <code>UseStatus</code> enumeration that specifies how the <code>Changeable</code> object behaves when it is "used." A <code>Changeable</code> object is considered used in the following situations: the object is set into a <code>Property System</code> property, the object is used as a sub-object in a complex <code>Changeable</code> object, or the object is used in a <code>DrawingContext</code> command. Inherited from <code>Changeable</code> .
Transform	Gets or sets the <code>Transform</code> object applied to a <code>Geometry</code> . Inherited from <code>Geometry</code> .
UIContext	Gets the <code>UIContext</code> of the current object. The <code>UIContext</code> is used for maintaining thread safety. Inherited from <code>Changeable</code> .

This example demonstrates how to draw shapes using the `Geometry`, `PathFigure`, and `PathSegment` classes. In this example, several shapes are drawn using `Geometry` objects and are displayed using a `Path` element.

There is a `Geometry` class for each basic geometric shape: `LineGeometry`, `EllipseGeometry`, and `RectangleGeometry`. Complex shapes, such as polygons and shapes with curved segments, may be created using a `PathGeometry`.

In the following example, a `LineGeometry`, `EllipseGeometry`, and a `RectangleGeometry` object are used to create a line, an ellipse, and a rectangle.

```

// C#
MSAvalon.Windows.Media.LineGeometry myLineGeometry =
    new LineGeometry(new Point(50, 50), new Point(300, 50));

MSAvalon.Windows.Media.EllipseGeometry myEllipseGeometry =
    new EllipseGeometry(new Point(440, 100), 40, 75);

MSAvalon.Windows.Media.RectangleGeometry myRectangleGeometry =
    new RectangleGeometry(new Rect(new Point(400, 225), new Size(100, 50)));

' VB .NET
Dim myLineGeometry As new MSAvalon.Windows.Media.LineGeometry( _
    new MSAvalon.Windows.Point(50, 50), new MSAvalon.Windows.Point(300, 50))
Dim myEllipseGeometry As new MSAvalon.Windows.Media.EllipseGeometry( _
    new MSAvalon.Windows.Point(440, 100), 40, 75)
Dim myRectangleGeometry As new MSAvalon.Windows.Media.RectangleGeometry( _
    new MSAvalon.Windows.Rect(new MSAvalon.Windows.Point(400,225), _
    new MSAvalon.Windows.Size(100,50)))

```

PathGeometry objects can be used to create complex shapes, such as arcs and curves. PathGeometry objects are comprised of one or more PathFigure objects; each PathFigure represents a different "figure" or shape. Each PathFigure is itself comprised of one or more PathSegment objects, each representing a connected portion of the figure or shape. Segment types include the following: LineSegment, BezierSegment, and ArcSegment.

In the following code, a PathGeometry and a PathFigure are created, and several segments are added to the PathFigure to form a shape. There are several ways to add segments to a PathFigure; you can use the PathFigure object's "draw segment" commands to automatically create new segments and add them to the figure, or you can explicitly create segments and add them manually using the PathFigure object's Segments property or AddSegment method. This example shows both ways of adding segments to a figure.

The first segment of a PathFigure must be a StartSegment. The StartSegment may be added by creating a new StartSegment and adding it to the PathFigure, or it can be added using the PathFigure object's StartAt method. The following code demonstrates adding a StartSegment using the StartAt method. The start point is set to (200,50).

```

// C#
MSAvalon.Windows.Media.PathGeometry myPathGeometry = new PathGeometry();
MSAvalon.Windows.Media.PathFigure myPathFigure = new PathFigure();

// PathFigure objects must have a defined start point before
// other segments can be added.
myPathFigure.StartAt(new Point(200,50));

' VB .NET
Dim myPathGeometry As new MSAvalon.Windows.Media.PathGeometry()
Dim myPathFigure As new MSAvalon.Windows.Media.PathFigure()

' PathFigure objects must have a defined start point before
' other segments can be added.
myPathFigure.StartAt(new MSAvalon.Windows.Point(200,50))

```

In the following code, the BezierTo method is used to create three Bézier curves. A fourth curve is created by explicitly creating a BezierSegment and adding it to myPathFigure using the AddSegment

method. After the segments are added to the PathFigure (myPathFigure), the PathFigure is added to the PathGeometry.

```
// C#
myPathFigure.BezierTo(
    new Point(400, 100), new Point(400, 200), new Point(200, 300));

myPathFigure.BezierTo(
    new Point(400, 300), new Point(400, 100), new Point(200, 50));

myPathFigure.BezierTo(
    new Point(0, 100), new Point(0, 200), new Point(200,300));

MSAvalon.Windows.Media.BezierSegment myBezierSegment =
    new BezierSegment(new Point(0, 300), new Point(0, 100), new Point(200, 50), true);

myPathFigure.AddSegment(myBezierSegment);

myPathGeometry.Figures.Add(myPathFigure);

' VB .NET
myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 100), _
    new MSAvalon.Windows.Point(400, 200), new MSAvalon.Windows.Point(200, 300))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(400, 300), _
    new MSAvalon.Windows.Point(400, 100), new MSAvalon.Windows.Point(200, 50))

myPathFigure.BezierTo(new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(0, 200), new MSAvalon.Windows.Point(200,300))

Dim myBezierSegment As new BezierSegment( _
    new MSAvalon.Windows.Point(0, 300), new MSAvalon.Windows.Point(0, 100), _
    new MSAvalon.Windows.Point(200, 50), true)

myPathFigure.AddSegment(myBezierSegment)

' Add the PathFigure to the PathGeometry
myPathGeometry.Figures.Add(myPathFigure)
```

In the final code example, the geometries are added to a GeometryCollection, and the GeometryCollection is used to set the Path element's Data property. Had there been only one geometry, it could have been used to set the Data property directly, without the GeometryCollection.

```
// C#
GeometryCollection myGeometryCollection = new GeometryCollection();
myGeometryCollection.Add(myLineGeometry);
myGeometryCollection.Add(myEllipseGeometry);
myGeometryCollection.Add(myRectangleGeometry);
myGeometryCollection.Add(myPathGeometry);

Path myPath = new Path();
myPath.Data = myGeometryCollection;

// Set the outline and the fill of the Path element.
myPath.Stroke = Brushes.Blue;
myPath.StrokeThickness = new Length(5);
SolidColorBrush solidFill = new SolidColorBrush();
myPath.Fill = new RadialGradientBrush(Colors.Orange, Colors.Red);
```

```
// Add the Path element to a Canvas.  
myCanvas.Children.Add(myPath);
```

```
' VB .NET
```

```
Dim myGeometryCollection As new MS Avalon.Windows.Media.GeometryCollection()  
myGeometryCollection.Add(myLineGeometry)  
myGeometryCollection.Add(myEllipseGeometry)  
myGeometryCollection.Add(myRectangleGeometry)  
myGeometryCollection.Add(myPathGeometry)
```

```
Dim myPath As new Path()  
myPath.Data = myGeometryCollection
```

```
' Set the outline and the fill of the Path element.  
myPath.Stroke = MS Avalon.Windows.Media.Brushes.Blue  
myPath.StrokeThickness = new MS Avalon.Windows.Length(5)  
Dim solidFill As new MS Avalon.Windows.Media.SolidColorBrush()  
myPath.Fill = new RadialGradientBrush(MS Avalon.Windows.Media.Colors.Orange, _  
    MS Avalon.Windows.Media.Colors.Red)
```

```
' Add the Path element to a Canvas.  
myCanvas.Children.Add(myPath)
```

Geometry objects may also be rendered using the `DrawingContext`, which supplies a `DrawGeometry` method that may be used to render Geometry objects. Geometry objects may also be used for clipping and hit-testing.

This example demonstrates how to draw shapes using the Geometry and Path elements in "Longhorn" markup language (code-named "XAML").

In the following example, a Path is drawn on a Canvas. Several Geometry elements are assigned to the Path element's Data attribute.

```
<Canvas ID="root"  
    Background="White"  
    xmlns="http://schemas.microsoft.com/2003/xaml">  
  
    <Path ID="myPath"  
        Fill="Blue"  
        Stroke="Black"  
        StrokeThickness="5">  
  
        <Path.Data>  
            <GeometryCollection>  
  
                <LineGeometry StartPoint="50,50" EndPoint="300,50"/>  
                <EllipseGeometry Center="440, 100" RadiusX="40" RadiusY="75"/>  
                <RectangleGeometry >  
                    <RectangleGeometry.Rect>  
                        <Rect X="400" Y="225" Width="100" Height="50"/>  
                    </RectangleGeometry.Rect>  
                </RectangleGeometry>  
  
            </PathGeometry>  
            <PathGeometry.Figures>  
                <PathFigureCollection>
```

```

<PathFigure>
  <PathFigure.Segments>
    <PathSegmentCollection>
      <StartSegment Point="400,100"/>
      <BezierSegment Point1="400,100" Point2="400,200" Point3="200,300"/>
      <BezierSegment Point1="400,300" Point2="400,100" Point3="200,50"/>
      <BezierSegment Point1="0,100" Point2="0,200" Point3="200,300"/>
      <BezierSegment Point1="0,300" Point2="0,100" Point3="200,50"/>
    </PathSegmentCollection>
  </PathFigure.Segments>
</PathFigure>
</PathFigureCollection>
</PathGeometry.Figures>
</PathGeometry>

```

```

</GeometryCollection>

```

```

</Path.Data>
</Path>

```

```

</Canvas>

```

In the previous example, the PathFigure object, one of the shapes drawn inside the Path element, contains a StartSegment but no CloseSegment; if a CloseSegment were added to the figure, a line would be drawn from the last segment in the collection back to the starting point of the figure.

RetainedVisual Class

Definition: RetainedVisual

Method	Description
ClearValue	Clears the local value of a property Inherited from DependencyObject.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Releases all resources held by the Visual object. Inherited from Visual.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetLocalValueEnumerator	Create a local value enumerator for this instance Inherited from DependencyObject.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Retrieve the value of a property Inherited from DependencyObject.
HitTestCore	HitTestCore implements precise hit testing against render contents
InvalidateProperty	Invalidates a property Inherited from DependencyObject.
InvalidateVisual	InvalidateVisual.
IVisual.FindCommonVisualAncestor	Inherited from Visual.
IVisual.HitTest	Inherited from Visual.
IVisual.IsAncestorOf	Inherited from Visual.

IVisual.IsDescendantOf	Inherited from Visual.
IVisual.TransformFromAncestor	Inherited from Visual.
IVisual.TransformFromDescendant	Inherited from Visual.
IVisual.TransformFromVisual	Inherited from Visual.
IVisual.TransformToAncestor	Inherited from Visual.
IVisual.TransformToDescendant	Inherited from Visual.
IVisual.TransformToVisual	Inherited from Visual.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
OnDelayedInvalidate	TODO: Left over from WCP FastBuild, determine relevance in future version of FastBuild Inherited from DependencyObject.
OnPropertyInvalidated	Notification that a specified property has been invalidated Inherited from DependencyObject.
ReadLocalValue	Retrieve the local value of a property (if set) Inherited from DependencyObject.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
RenderOpen	RendeOpen opens the RetainedVisual for rendering.
RetainedVisual	RetainedVisual ctor.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
SetValue	Sets the local value of a property Inherited from DependencyObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateProperty	Retrieve the value of a property (for use by native cache backed custom get accessors) Inherited from DependencyObject.
ValidatePropertyCore	Allows subclasses to participate in property value computation Inherited from DependencyObject.

Property	Description
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
DependencyObjectType	Returns the DType that represents the CLR type of this instance Inherited from DependencyObject.
HitTestBounds	HitBounds returns the hit region bounding box for the current visual. Inherited from Visual.
IsDisposed	Gets a value that indicates whether the system has disposed of the Visual. Inherited from Visual.
RenderBounds	This property is only used if the RetainedVisual implements RetainedRender. If not this property will throw an InvalidOperationException. The implementer must set this property to the bounds of the ink drawn by his Render function in local coordinate space.

RotateTransform Class

Definition: Used to rotate an object about a specified point in the two-dimensional x-y plane.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this Transform. Inherited from Transform.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this RotateTransform.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane. Inherited from Transform.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane. Inherited from Transform.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane. Inherited from Transform.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane. Inherited from Transform.
CreateTranslation	Creates a transformation used for translating in the x- and y-directions in a two-dimensional plane. Inherited from Transform.
DisableCore	Inherited from Transform.
Dispose	Dispose resources associated with transform. Inherited from Transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Transform.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Transform.
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base values set to the current animated values. Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this RotateTransform that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Transform.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.

GetIsAnimating	Inherited from Transform.
GetIsOverridingBaseValue	Inherited from Transform.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Transform.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
RotateTransform	Initializes a new instance of the RotateTransform class. The rotation angle is measured in degrees clockwise. The default center of rotation is the origin.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimeline	Inherited from Transform.
SetDefaultParentTimelineCore	Inherited from Transform.
ToString	Not implemented. Use Object.ToString instead. Inherited from Transform.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.

Angle	Gets or sets the angle of rotation, measured in degrees clockwise.
AngleAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's Angle property.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Center	Gets or sets the center point of rotation.
CenterAnimations	Gets or sets a collection of PointModifier objects that animate the transformation's Center property.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the transformation has animations.
Identity	Identity transformation. Inherited from Transform.
IsAnimating	Returns true if the transformation contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Value	Gets the current rotation transformation as a Matrix object.

This example in "Longhorn" markup language (code-named "XAML") shows how to create a spinning Rectangle.

```

    <DockPanel xmlns="http://schemas.microsoft.com/2003/xaml">

<TransformDecorator>
  <TransformDecorator.Transform>
    <RotateTransform Center="100,100">
      <RotateTransform.AngleAnimations>

        <DoubleAnimation From="0" To="360" RepeatCount="30" Begin="1"
          Duration="4" />

      </RotateTransform.AngleAnimations>
    </RotateTransform>
  </TransformDecorator.Transform>

  <Rectangle RectangleTop="100" RectangleLeft="100" Fill="blue"
    RectangleWidth="50" RectangleHeight="50" Stroke="black"
    StrokeThickness="5" />

</TransformDecorator>

```

</DockPanel>

In the previous example, the `DoubleAnimationCollection` tag, `<DoubleAnimationCollection>`, is omitted when animating the transformation's angle. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `AngleAnimations` property in the previous example—you may omit the animation collection tag. However, when you animate a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

This example specifies rotation transforms on several shapes. The `Transform` class enables you to reposition and apply rendering changes to shape elements such as `Rectangle`, `Polyline`, and `Ellipse`. The `RotateTransform` class specifies a rotation of the coordinate system. By default, the coordinate system is rotated about (0,0), the default origin, by *r* degrees. The following code provides some examples of different rotations, some in conjunction with translation.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml"
  xmlns:def="Definition">

  <!-- No transform -->
  <Polyline ID="box1"
    Stroke="Black"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10" />

  <!-- Rotate about a set origin, then translate -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection>
        <RotateTransform Angle="90" />
        <TranslateTransform X="100" Y="100" />
      </TransformCollection>
    </TransformDecorator.Transform>
  </TransformDecorator>
  <Polyline ID="box2"
    Stroke="Blue"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
  </Polyline>
  </TransformDecorator>

  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection>
        <RotateTransform Angle="-45" />
        <TranslateTransform X="100" Y="100" />
      </TransformCollection>
    </TransformDecorator.Transform>
  </TransformDecorator>
  <Polyline ID="box3"
    Stroke="Orange"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
  </Polyline>
  </TransformDecorator>
```

```

<TransformDecorator AffectsLayout="false">
<TransformDecorator.Transform>
    <TransformCollection>

        <RotateTransform Angle="45" />
        <TranslateTransform X="100" Y="100" />

    </TransformCollection>
</TransformDecorator.Transform>
<Polyline ID="box4"
    Stroke="Green"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
</Polyline>
</TransformDecorator>

<!-- Translate, then rotate -->
<TransformDecorator AffectsLayout="false">
<TransformDecorator.Transform>
    <TransformCollection>

        <TranslateTransform X="200" Y="200" />
        <RotateTransform Angle="15" />

    </TransformCollection>
</TransformDecorator.Transform>
<Polyline ID="box5"
    Stroke="Cyan"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
</Polyline>
</TransformDecorator>
</Canvas>

```

When using rotation transformations, keep in mind that the transformation rotates the coordinate system for a particular element. This might apply effects that you should anticipate when working out the screen coordinates for elements. Depending on an element's position with respect to the origin, the effect of the rotation might not be to rotate it "in place." For an element positioned 200 units from 0 along the x-axis, for instance, a rotation of 30 degrees has the effect of swinging the element 30 degrees along a circle with radius 200, drawn around the origin.

The Center attribute on the RotateTransform element allows you to set an origin about which the shape will be rotated. Note that in the preceding example, the start points of the blue, orange, and green Polyline elements are all located at (100,100) because the Center attribute of those elements is (100,100). The cyan Polyline, however, was rotated 15 degrees on an arc from (200,200) because its TranslateTransform was applied first.

This example uses a RotateTransform object to rotate a Shape element. To scale an element, use a ScaleTransform object to specify the transformation, and a TransformDecorator object to apply the transformation. Instead of adding the element to the Canvas element or other Panel element, add the element as child of the TransformDecorator and add the TransformDecorator to the Panel.

The RotateTransform object specifies a rotation of the coordinate system. By default, the coordinate system is rotated about (0,0), the default origin, by r degrees. The following code shows the two shapes. The first shape is not rotated. The second shape is rotated by 45 degrees around a RotateTransform of (110,110), the lower right corner of the shape. As a result, the shape appears as though it was rotated about its corner.

```

// C#
// Create a canvas to contain the shapes.

```

```

Canvas myCanvas = new Canvas();

// Create a PointCollection to contain the points of the Polygon shapes.
PointCollection myPointCollection = new PointCollection();
myPointCollection.Add(new Point(60, 60));
myPointCollection.Add(new Point(70, 70));
myPointCollection.Add(new Point(70, 110));
myPointCollection.Add(new Point(110, 110));
myPointCollection.Add(new Point(110, 70));
myPointCollection.Add(new Point(70, 70));

// Create the first Polygon and add it to the Canvas.
Polygon box1 = new Polygon();
box1.Points = myPointCollection;
box1.Stroke = Brushes.Black;
box1.StrokeThickness = new Length(5);
myCanvas.Children.Add(box1);

// Create the second Polygon. This polygon contains the same points as the
// first, but is rotated.
Polygon box2 = new Polygon();
box2.Points = myPointCollection;
box2.Stroke = Brushes.Blue;
box2.StrokeThickness = new Length(5);

// Create a TransformDecorator to transform the shape.
TransformDecorator transformer = new TransformDecorator();
transformer.AffectsLayout = false;

// Create the rotation transformation.
RotateTransform myRotateTransform = new RotateTransform(45, new Point(110, 110));

// Set the TransformDecorator.Transform property.
transformer.Transform = myRotateTransform;

// Add box2 as a child of the TransformDecorator and add the TransformDecorator
// to the Canvas.
transformer.Child = box2;
myCanvas.Children.Add(transformer);

' VB .NET
' Create a PointCollection to contain the points of the
' Polygon shapes.
Dim myPointCollection As MS Avalon.Windows.Media.PointCollection
myPointCollection = new MS Avalon.Windows.Media.PointCollection
myPointCollection.Add(new MS Avalon.Windows.Point(60, 60))
myPointCollection.Add(new MS Avalon.Windows.Point(70, 70))
myPointCollection.Add(new MS Avalon.Windows.Point(70, 110))
myPointCollection.Add(new MS Avalon.Windows.Point(110, 110))
myPointCollection.Add(new MS Avalon.Windows.Point(110, 70))
myPointCollection.Add(new MS Avalon.Windows.Point(70, 70))

' Create the first Polygon and add it to the Canvas.
Dim box1 As MS Avalon.Windows.Shapes.Polygon
box1 = new MS Avalon.Windows.Shapes.Polygon

```

```

box1.Points = myPointCollection
box1.Stroke = Brushes.Black
box1.StrokeThickness = new MS Avalon.Windows.Length(5)
myCanvas.Children.Add(box1)

' Create the second Polygon. This polygon contains the same
' points as the first, but is rotated.
Dim box2 As MS Avalon.Windows.Shapes.Polygon
box2 = new MS Avalon.Windows.Shapes.Polygon
box2.Points = myPointCollection
box2.Stroke = Brushes.Blue
box2.StrokeThickness = new MS Avalon.Windows.Length(5)

' Create a TransformDecorator to transform the shape.
Dim transformer As MS Avalon.Windows.Controls.TransformDecorator
transformer = new MS Avalon.Windows.Controls.TransformDecorator
transformer.AffectsLayout = false

' Create the scale transformation.
Dim myRotateTransform As _
    new MS Avalon.Windows.Media.RotateTransform(45, _
    new MS Avalon.Windows.Point(110,110))

'Set the TransformDecorator.Transform property
transformer.Transform = myRotateTransform

'Add box2 as a child of the TransformDecorator and add
'the TransformDecorator to the Canvas.
transformer.Child = box2
myCanvas.Children.Add(transformer)

```

When using rotation transformations, keep in mind that the transformation rotates the coordinate system for a particular element. This might apply effects that you should anticipate when working out the screen coordinates for elements. Depending on an element's position with respect to the origin, the effect of the rotation might not be to rotate it "in place." For example, for an element positioned 200 units from 0 along the x-axis a rotation of 30 degrees has the effect of swinging the element 30 degrees along a circle with radius 200, drawn around the origin.

Although the previous example shows how to rotate a Shape element, the process is the same for other elements, including Button and Panel elements.

ScaleTransform Class

Definition: Scales an object in the two-dimensional x-y plane, starting from a defined center point. Scale factors are defined in x- and y-directions from this center point.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this ScaleTransform.

Copy	Creates a copy of this Transform. Inherited from Transform.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane. Inherited from Transform.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane. Inherited from Transform.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane. Inherited from Transform.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane. Inherited from Transform.
CreateTranslation	Creates a transformation used for translating in the x- and y-directions in a two-dimensional plane. Inherited from Transform.
DisableCore	Inherited from Transform.
Dispose	Dispose resources associated with transform. Inherited from Transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Transform.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this ScaleTransform that represents its current state.
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base values set to the current animated values. Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Transform.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Transform.
GetIsOverridingBaseValue	Inherited from Transform.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Transform.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified

OnChanged	Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> . Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
PropagateEventHandler	
ReadPreamble	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
ReferenceEquals	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
ScaleTransform	Initializes a new instance of the <code>ScaleTransform</code> class.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this <code>Changeable</code> which are inheriting their time parent. Inherited from <code>Animatable</code> .
SetDefaultParentTimeline	Inherited from <code>Transform</code> .
SetDefaultParentTimelineCore	Inherited from <code>Transform</code> .
ToString	Not implemented. Use <code>Object.ToString</code> instead. Inherited from <code>Transform</code> .
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
WritePostscript	Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code> .
WritePreamble	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from <code>Changeable</code> .

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
CanMakeUnchangeable	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
Center	Gets or sets the center point of the scaling operation. Scale factors are defined in x- and y-directions from this center point.
CenterAnimations	Gets or sets a collection of <code>PointModifier</code> objects that animate the transformation's <code>Center</code> property.
DefaultParentTimeline	The current parent <code>Timeline</code> associated with this <code>Animatable</code> . This will be the <code>Timeline</code> set to the <code>ParentTimeline</code> property of this <code>Animatable</code> if one has been set and if not, the <code>Timeline</code> last passed into the <code>SetDefaultParentTimeline</code> method. Inherited from <code>Animatable</code> .
HasAnimations	Gets a <code>Boolean</code> that indicates whether the transformation has animations.
Identity	Identity transformation. Inherited from <code>Transform</code> .

IsAnimating	Returns true if the transformation contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
ScaleX	Gets or sets the scale factor in the x-direction.
ScaleXAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's ScaleX property.
ScaleY	Gets or sets the scale factor in the y-direction.
ScaleYAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's ScaleY property.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Value	Gets the current scaling transformation as a Matrix object.

This example specifies scale transformations on several shapes. The ScaleTransform class enables you to reposition and apply rendering changes to shape elements like Rectangle, Polygon, and Ellipse. The ScaleX and ScaleY properties resize an element according to the factor you specify. For example, setting ScaleX to 1.5 resizes the element's X-axis value at 150 percent.

The following example demonstrates a polygon scaled in several ways. The first polygon (black) reflects no transformation. The second polygon (blue) is scaled at 50 percent in Y, and the third polygon (red) is scaled at 50 percent in X. Two of the polygons also include translation transformations for repositioning. Note that the scale transformation affects the element's stroke-thickness as well as its size.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <!-- No transform specified -->
  <Polygon ID="star1"
    Stroke="black"
    StrokeThickness="2.0"
    Points="176.5,50 189.2,155.003 286.485,113.5 201.9,177 286.485,240.5
    189.2,198.997 176.5,304 163.8,198.997 66.5148,240.5 151.1,177 66.5148,113.5
    163.8,155.003">
  </Polygon>

  <!-- Scaled in Y -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection>
        <ScaleTransform ScaleX="1" ScaleY="0.5" />
        <TranslateTransform X="100" Y="0" />
      </TransformCollection>
    </TransformDecorator.Transform>
  <Polygon ID="star2"
    Stroke="blue"
    StrokeThickness="2.0"
```



```

Points="176.5,50 189.2,155.003 286.485,113.5 201.9,177 286.485,240.5
189.2,198.997 176.5,304 163.8,198.997 66.5148,240.5 151.1,177 66.5148,113.5
163.8,155.003" />
</TransformDecorator>

<!-- Scaled in X -->
<TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
        <TransformCollection>
            <ScaleTransform ScaleX="0.5" ScaleY="1" />
            <TranslateTransform X="0" Y="100" />
        </TransformCollection>
    </TransformDecorator.Transform>
    <Polygon ID="star3"
    Stroke="red"
    StrokeThickness="2.0"
    Points="176.5,50 189.2,155.003 286.485,113.5 201.9,177 286.485,240.5
    189.2,198.997 176.5,304 163.8,198.997 66.5148,240.5 151.1,177 66.5148,113.5
    163.8,155.003" />
</TransformDecorator>

</Canvas>

```

This example specifies scale transformations on several shapes. To transform an element, use a `TransformDecorator` and set its `Transform` to the transformation you wish to apply. Set the element to transform as the `TransformDecorator` object's `Child`, and add the `TransformDecorator` to a panel. The `ScaleTransform` class, one of the transformations you can use with a `TransformDecorator`, enables you to reposition and apply rendering changes elements like `Rectangle`, `Polygon`, and `Button`. The `ScaleX` and `ScaleY` properties resize an element according to the factor you specify. For example, setting `ScaleX` to 1.5 resizes the element's x-axis value at 150 percent. The following example demonstrates a polygon scaled in several ways. The first polygon (black) reflects no transformation.

```

// #C
// Create a canvas to contain the shapes.
Canvas myCanvas = new Canvas();

// Create a PointCollection to contain the points of the
// Polygon shapes.
PointCollection myPointCollection = new PointCollection();
myPointCollection.Add(new Point(176.5, 50));
myPointCollection.Add(new Point(189, 155));
myPointCollection.Add(new Point(286, 113));
myPointCollection.Add(new Point(201, 177));
myPointCollection.Add(new Point(286, 240));
myPointCollection.Add(new Point(189, 198));
myPointCollection.Add(new Point(176, 304));
myPointCollection.Add(new Point(163, 198));
myPointCollection.Add(new Point(66, 240));
myPointCollection.Add(new Point(151, 177));
myPointCollection.Add(new Point(66, 113));
myPointCollection.Add(new Point(163, 155));

// Create the first Polygon.
Polygon firstPolygon = new Polygon();
firstPolygon.Points = myPointCollection;

```

```

firstPolygon.Stroke = Brushes.Black;
firstPolygon.StrokeThickness = new Length(2);
myCanvas.Children.Add(firstPolygon);

```

```

' VB .NET
' Create a PointCollection to contain the points of the
' Polygon shapes.
Dim myPointCollection As new MS Avalon.Windows.Media.PointCollection
myPointCollection.Add(new MS Avalon.Windows.Point(176.5, 50))
myPointCollection.Add(new MS Avalon.Windows.Point(189, 155))
myPointCollection.Add(new MS Avalon.Windows.Point(286, 113))
myPointCollection.Add(new MS Avalon.Windows.Point(201, 177))
myPointCollection.Add(new MS Avalon.Windows.Point(286, 240))
myPointCollection.Add(new MS Avalon.Windows.Point(189, 198))
myPointCollection.Add(new MS Avalon.Windows.Point(176, 304))
myPointCollection.Add(new MS Avalon.Windows.Point(163, 198))
myPointCollection.Add(new MS Avalon.Windows.Point(66, 240))
myPointCollection.Add(new MS Avalon.Windows.Point(151, 177))
myPointCollection.Add(new MS Avalon.Windows.Point(66, 113))
myPointCollection.Add(new MS Avalon.Windows.Point(163, 155))

' Create the first Polygon.
Dim firstPolygon As new MS Avalon.Windows.Shapes.Polygon
firstPolygon.Points = myPointCollection
firstPolygon.Stroke = MS Avalon.Windows.Media.Brushes.Black
firstPolygon.StrokeThickness = new MS Avalon.Windows.Length(2)
myCanvas.Children.Add(firstPolygon)

```

The second polygon (blue) is scaled at 50 percent in y and translated by 150 pixels. Because there are multiple transformations, a TransformCollection object is used to contain the transformations and set the value of the TransformDecorator. Note that the scale transformation affects the element's stroke thickness as well as its size. In order to for the TranslateTransform to render, the AffectsLayout property of the TransformDecorator must be set to false.

```

// C#
// Create the second Polygon. This polygon contains the same
// points as the first, but is scaled and translated.
Polygon secondPolygon = new Polygon();
secondPolygon.Points = myPointCollection;
secondPolygon.Stroke = Brushes.Blue;
secondPolygon.StrokeThickness = new Length(2);

// Add a semi-transparent gradient fill to make the shape stand out.
RadialGradientBrush myGradient =
    new RadialGradientBrush(Colors.Blue, Colors.LimeGreen);
myGradient.Opacity = 0.4;
secondPolygon.Fill = myGradient;

// Create a TransformDecorator to transform secondPolygon.
TransformDecorator transformer = new TransformDecorator();
transformer.AffectsLayout = false;

// Create the scale and translate transformations.

```

```
ScaleTransform myScaleTransform = new ScaleTransform(1, 0.5);
TranslateTransform myTranslateTransform = new TranslateTransform(150, 0);
```

```
//Create a collection to contain the transformations.
TransformCollection transformations = new TransformCollection();
```

```
transformations.Add(myScaleTransform);
transformations.Add(myTranslateTransform);
transformer.Transform = transformations;
```

```
// Add secondPolygon as a child of the TransformDecorator.
transformer.Child = secondPolygon;
```

```
// Add the decorator to the Canvas.
myCanvas.Children.Add(transformer);
```

```
' VB .NET
' Create the second Polygon. This polygon contains the same
' points as the first, but is scaled and translated.
Dim secondPolygon As new MS Avalon.Windows.Shapes.Polygon
secondPolygon.Points = myPointCollection
secondPolygon.Stroke = MS Avalon.Windows.Media.Brushes.Blue
secondPolygon.StrokeThickness = new MS Avalon.Windows.Length(2)

' Add a semi-transparent gradient fill to make the shape stand out.
Dim myGradient As new MS Avalon.Windows.Media.RadialGradientBrush( _
    Colors.Blue, Colors.LimeGreen)
myGradient.Opacity = 0.4
secondPolygon.Fill = myGradient

' Create a TransformDecorator to transform secondPolygon.
Dim transformer As new MS Avalon.Windows.Controls.TransformDecorator
transformer.AffectsLayout = false

' Create the scale and translate transformations.
Dim myScaleTransform As new MS Avalon.Windows.Media.ScaleTransform(1, 0.5)
Dim myTranslateTransform As new MS Avalon.Windows.Media.TranslateTransform(150, 0)

' Create a collection to contain the transformations.
Dim transformations As new MS Avalon.Windows.Media.TransformCollection()

transformations.Add(myScaleTransform)
transformations.Add(myTranslateTransform)
transformer.Transform = transformations

' Add secondPolygon as a child of the TransformDecorator.
transformer.Child = secondPolygon

' Add the decorator to the Canvas.
myCanvas.Children.Add(transformer)
```

This example uses classes from the `MSAvalon.Windows`, `MSAvalon.Windows.Media`, and `MSAvalon.Windows.Controls` namespaces.

This example specifies scale transforms with a center point other than (0,0). To scale an element, use a `ScaleTransform` to specify the transformation, and a `TransformDecorator` to apply the transformation. Instead of adding the element to the `Canvas` or other `Panel`, add the element as child of the `TransformDecorator` and add the `TransformDecorator` to the `Panel`.

The following example shows the effect of scaling on `Rectangle` elements. The first rectangle is not scaled. The second rectangle is scaled vertically by 2 from a Center of (0,0), the default center. Although the second rectangle was originally 50 pixels from the top of the canvas (before the transformation), the rectangle looks as though it were moved down another 50 pixels after the transformation. The third rectangle is vertically scaled by 2 from a center of (0,50). Like the second rectangle, the third rectangle is vertically doubled and its distance from the top of the canvas is doubled as well, but specifying the scale center of (0,50) has the effect of moving the rectangle 50 pixels up, placing its top edge back in alignment with the first rectangle.

```
// C#
// Create a canvas to contain the shapes.
Canvas myCanvas = new Canvas();

// Create and set the first shape, and add it to the Canvas.
Rectangle firstRectangle = new Rectangle();
firstRectangle.RectangleTop = new Length(50);
firstRectangle.RectangleLeft = new Length(50);
firstRectangle.RectangleHeight = new Length(50);
firstRectangle.RectangleWidth = new Length(50);
firstRectangle.Fill = Brushes.Red;
firstRectangle.Stroke = Brushes.Black;
myCanvas.Children.Add(firstRectangle);

// Create and set the second shape.
Rectangle secondRectangle = new Rectangle();
secondRectangle.RectangleTop = new Length(50);
secondRectangle.RectangleLeft = new Length(150);
secondRectangle.RectangleHeight = new Length(50);
secondRectangle.RectangleWidth = new Length(50);
secondRectangle.Fill = Brushes.Yellow;
secondRectangle.Stroke = Brushes.Black;

// Create a TransformDecorator to apply the transformation to the shape.
TransformDecorator transformation = new TransformDecorator();
transformation.AffectsLayout = false;

// Create a scale transform which doubles the length of the shape.
// No scale is set, so it defaults to (0,0).
ScaleTransform myScaleTransform = new ScaleTransform();
myScaleTransform.ScaleY = 2.0;

// Assign the ScaleTransform to the TransformDecorator
transformation.Transform = myScaleTransform;

// Make the shape the child of the TransformDecorator and add the
// TransformDecorator (which now contains the shape) to the Canvas.
transformation.Child = secondRectangle;
myCanvas.Children.Add(transformation);

// Create and set the third shape.
Rectangle thirdRectangle = new Rectangle();
```

```

thirdRectangle.RectangleTop = new Length(50);
thirdRectangle.RectangleLeft = new Length(250);
thirdRectangle.RectangleHeight = new Length(50);
thirdRectangle.RectangleWidth = new Length(50);
thirdRectangle.Fill = Brushes.Blue;
thirdRectangle.Stroke = Brushes.Black;

// Create a new scale transform. This time the
// transform's center is set to (0,50).
myScaleTransform = new ScaleTransform();
myScaleTransform.Center = new Point(0, 50);
myScaleTransform.ScaleY = 2.0;

// Create another TransformDecorator to contain the third shape.
// Assign the shape and the transformation to the TransformDecorator,
// and add the TransformDecorator to the Canvas.
TransformDecorator secondTransformation = new TransformDecorator();
secondTransformation.AffectsLayout = false;
secondTransformation.Transform = myScaleTransform;
secondTransformation.Child = thirdRectangle;
myCanvas.Children.Add(secondTransformation);

' VB .NET
' Create and set the first shape, and add it to the Canvas.
Dim firstRectangle As MS Avalon.Windows.Shapes.Rectangle
firstRectangle = new MS Avalon.Windows.Shapes.Rectangle
firstRectangle.RectangleTop = new MS Avalon.Windows.Length(50)
firstRectangle.RectangleLeft = new MS Avalon.Windows.Length(50)
firstRectangle.RectangleHeight = new MS Avalon.Windows.Length(50)
firstRectangle.RectangleWidth = new MS Avalon.Windows.Length(50)
firstRectangle.Fill = MS Avalon.Windows.Media.Brushes.Red
firstRectangle.Stroke = MS Avalon.Windows.Media.Brushes.Black
myCanvas.Children.Add(firstRectangle)

' Create and set the second shape.
Dim secondRectangle As MS Avalon.Windows.Shapes.Rectangle
secondRectangle = new MS Avalon.Windows.Shapes.Rectangle
secondRectangle.RectangleTop = new MS Avalon.Windows.Length(50)
secondRectangle.RectangleLeft = new MS Avalon.Windows.Length(150)
secondRectangle.RectangleHeight = new MS Avalon.Windows.Length(50)
secondRectangle.RectangleWidth = new MS Avalon.Windows.Length(50)
secondRectangle.Fill = MS Avalon.Windows.Media.Brushes.Yellow
secondRectangle.Stroke = MS Avalon.Windows.Media.Brushes.Black

' Create a TransformDecorator to apply the transformation to the shape.
Dim transformation As new MS Avalon.Windows.Controls.TransformDecorator
transformation.AffectsLayout = false

' Create a scale transform which doubles the length of the shape.
' No scale is set, so it defaults to (0,0).
Dim myScaleTransform As new MS Avalon.Windows.Media.ScaleTransform
myScaleTransform.ScaleY = 2.0

' Assign the ScaleTransform to the TransformDecorator

```

```
transformation.Transform = myScaleTransform
```

```
' Make the shape the child of the TransformDecorator and add the  
' TransformDecorator (which now contains the shape) to the Canvas.  
transformation.Child = secondRectangle  
myCanvas.Children.Add(transformation)
```

```
' Create and set the third shape.  
Dim thirdRectangle As new MSAvalon.Windows.Shapes.Rectangle  
thirdRectangle.RectangleTop = new MSAvalon.Windows.Length(50)  
thirdRectangle.RectangleLeft = new MSAvalon.Windows.Length(250)  
thirdRectangle.RectangleHeight = new MSAvalon.Windows.Length(50)  
thirdRectangle.RectangleWidth = new MSAvalon.Windows.Length(50)  
thirdRectangle.Fill = MSAvalon.Windows.Media.Brushes.Blue  
thirdRectangle.Stroke = MSAvalon.Windows.Media.Brushes.Black
```

```
' Create a new scale transform. This time the  
' transform's center is set to (0,50).  
myScaleTransform = new MSAvalon.Windows.Media.ScaleTransform  
myScaleTransform.Center = new MSAvalon.Windows.Point(0, 50)  
myScaleTransform.ScaleY = 2.0
```

```
' Create another TransformDecorator to contain the third shape.  
' Assign the shape and the transformation to the TransformDecorator,  
' and add the TransformDecorator to the Canvas.
```

```
Dim secondTransformation As new MSAvalon.Windows.Controls.TransformDecorator  
secondTransformation.Transform = myScaleTransform  
secondTransformation.Child = thirdRectangle  
myCanvas.Children.Add(secondTransformation)
```

When you transform an element, it is the coordinate space containing the element that is actually transformed, not just the element itself. Because the TransformDecorator performs the transformation in its parent's coordinate space—the Canvas—the element will be moved. If you specify a y-axis scale of 2.0, not only is the element doubled, its distance from the top of the Canvas is doubled as well. There are several ways of moving the element back to its original position. You can move the element using a TranslateTransform, or you can specify the Center of the ScaleTransform, the more concise way of positioning the transformed element. The previous example moves the third rectangle by specifying the scale center.

This example uses classes from the MSAvalon.Windows, MSAvalon.Windows.Controls, MSAvalon.Windows.Media, and the MSAvalon.Windows.Shapes namespaces.

This example demonstrates how to animate the size of an element using "Longhorn" markup language (code-named "XAML"). There are multiple ways to animate the size of an element: directly animate the height and width attributes of the element, or apply an animated ScaleTransform to the element. In this example, two Rectangle elements are resized using these methods. One rectangle is resized by animating its RectangleWidth attribute and another is resized by animating a ScaleTransform applied to the rectangle. Each rectangle is filled with a pattern to highlight the differences between the two resizing methods. Initially, the two patterns look the same, but as the rectangles are resized, patterns change depending on how their containing rectangle is resized.

In the first example, a Rectangle element's RectangleWidth property is animated using a LengthAnimationCollection and a LengthAnimation. The LengthAnimation object in this example animates

the rectangle's `RectangleWidth` from its base value of 200 To a destination value of 600 over a Duration of 10 seconds.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Rectangle
    RectangleTop="20"
    RectangleLeft="20"
    RectangleWidth="200"
    RectangleHeight="150"
    Stroke="Red"
    StrokeThickness="5">

    <Rectangle.Fill>
      <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
        ImageSource="help.gif" TileMode="Tile"/>
    </Rectangle.Fill>

    <Rectangle.RectangleWidth>
      <!-- Animate the Rectangle's width: -->
      <LengthAnimationCollection>
        <LengthAnimation
          To="600" Duration="10" AutoReverse="true" RepeatCount="50" />
      </LengthAnimationCollection>
    </Rectangle.RectangleWidth>

  </Rectangle>
```

When the previous "XAML" is run, more of the pattern is exposed as the rectangle expands; however, the question marks that make up the pattern do not grow larger.

In the next example, a `TransformDecorator` is used to apply a `ScaleTransform` to a rectangle. A `DoubleAnimation` is used to animate the `ScaleTransform` object's `ScaleX` value using the `ScaleXAnimations` attribute. The `DoubleAnimation` animates the `ScaleX` value From 1 To a destination value of 3 over a Duration of 10 seconds. As a result, the rectangle's width is scaled from 100 percent (its starting size) to 300 percent over ten seconds.

```
<TransformDecorator AffectsLayout="False">
  <TransformDecorator.Transform>

    <!-- Use the ScaleTransform to enlarge the rectangle -->
    <ScaleTransform ScaleX="1" ScaleY="1">
      <ScaleTransform.ScaleXAnimations>
        <DoubleAnimation From="1" To="3" RepeatCount="30"
          AutoReverse="True" Begin="0" Duration="10" />
      </ScaleTransform.ScaleXAnimations>
    </ScaleTransform>

  </TransformDecorator.Transform>

  <Rectangle
    RectangleLeft="20"
    RectangleTop="200"
    RectangleWidth="200"
    RectangleHeight="150"
```

```

Stroke="Black"
StrokeThickness="3">
  <Rectangle.Fill>
    <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
      ImageSource="help.gif" TileMode="Tile"/>
    </Rectangle.Fill>
  </Rectangle>
</TransformDecorator>

```

</Canvas>

In the previous example, the `DoubleAnimationCollection` tag, `<DoubleAnimationCollection>`, is omitted when animating the transformation's scale factor. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ScaleXAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

When the second rectangle expands, the objects in the pattern also grow larger, unlike in the first rectangle. The pattern behaves this way because when you transform an element the entire element and its child elements are transformed. When you directly alter the size of an element, as in the case of the first rectangle, the element's children are not resized, unless their size and position are dependent on the size of their parent element.

For more information about animating properties, see [Animation in "Avalon"](#).

This example shows how to apply a `ScaleTransform` when an event occurs. The concept is the same for applying other types of transformations. See the `Transform` class for more information about the available types of transformations.

To apply a specific transformation to an element, you must create a `TransformDecorator` that you will add to that element. In markup, wrap the element with the `TransformDecorator`. When the event occurs, set the `TransformProperty` property of the `TransformDecorator` to the appropriate transformation, in this case a `ScaleTransform`.

```

<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml/"
  xmlns:def="Definition">

  <TransformDecorator ID="td1" AffectsLayout="false">
    <Button ID="Button1" MouseEnter="Enter" MouseLeave="Leave">Button</Button>
  </TransformDecorator>

  <def:Code>

    <![CDATA[
      Private Sub Enter(ByVal sender as object, ByVal args as MS Avalon.Windows.Input.MouseEventArgs)
        Dim scaler As new ScaleTransform(2,2)
        td1.Transform = scaler
      End Sub

      Private Sub Leave(ByVal sender as object, ByVal e as MS Avalon.Windows.Input.MouseEventArgs)
        Dim scaler As new ScaleTransform(1,1)
        td1.Transform = scaler
      End Sub
    ]]>
  </def:Code>

```


</def:Code>

</Canvas>

SkewTransform Class

Definition: Represents a two-dimensional skew.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this Transform. Inherited from Transform.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this SkewTransform.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane. Inherited from Transform.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane. Inherited from Transform.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane. Inherited from Transform.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane. Inherited from Transform.
CreateTranslation	Creates a transformation used for translating in the x- and y- directions in a two-dimensional plane. Inherited from Transform.
DisableCore	Inherited from Transform.
Dispose	Dispose resources associated with transform. Inherited from Transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Transform.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Transform.
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base values set to the current animated values. Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this SkewTransform that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that

GetHasAnimations	represents its current state. Inherited from Animatable. Inherited from Transform.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Transform.
GetIsOverridingBaseValue	Inherited from Transform.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Transform.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimeline	Inherited from Transform.
SetDefaultParentTimelineCore	Inherited from Transform.
SkewTransform	
ToString	Not implemented. Use Object.ToString instead. Inherited from Transform.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AngleX	Gets or sets the x-axis skew angle, measured in degrees counterclockwise from the y-axis.
AngleXAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's AngleX.
AngleY	Gets or sets the y-axis skew angle, measured in degrees counterclockwise from the x-axis.
AngleYAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's AngleY.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Center	Gets or sets the center point of the skew.
CenterAnimations	Gets or sets a collection of PointModifier objects that animate the transformation's Center position.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the transformation has animations.
Identity	Identity transformation. Inherited from Transform.
IsAnimating	Returns true if the transformation contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Value	Gets the current skewing transformation value as a Matrix.

SolidColorBrush Class

Definition: Represents a solid, uniform fill.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.

CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Note that the BrushType byte has not been read when this method is called.
DisableCore	Inherited from Brush.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Brush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Brush.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.

ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer
SetDefaultParentTimeline	Inherited from Brush.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from Brush.
SolidColorBrush	Initializes a new instance of the SolidColorBrush class.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Color	Gets or sets the color of the solid fill.
ColorAnimations	Gets or sets a collection of ColorModifier objects that animate the brush's Color property.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the brush has animations.

IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Gets a Boolean that indicates whether the SolidColorBrush is currently animated.
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

Use a SolidColorBrush to fill an area with a solid color. The Brushes class contains SolidColorBrush objects with predefined colors, such as Blue and Yellow.

The SolidColorBrush class represents only solid-color fills; see Brush for a variety of other, more complex, ways you can fill an area, such as gradients (LinearGradientBrush, RadialGradientBrush) and images (ImageBrush, NineGridBrush).

Color names are not case-sensitive.

This example uses the SolidColorBrush class in "Longhorn" markup language (code-named "XAML") to color areas such as the window background and the border and interior of shapes. The SolidColorBrush creates fills that are uniform in color—it can't create gradient or pattern fills. SolidColorBrush uses both predefined color values and hexadecimal color values. In this example, several Ellipse shapes are created with identical fills and outlines, but the fills and outlines are specified in different formats to demonstrate the versatility of the SolidColorBrush class.

In the following markup, a Canvas element is declared and its Background property is set to LightGray, one of the predefined colors.

```
<Canvas ID="root" xmlns="http://schemas.microsoft.com/2003/xaml"
  Background="LightGray">
```

In the next example, the Fill and Stroke properties of an Ellipse are set using ARGB notation. ARGB consists of a pound sign (#) and eight digits. The pound sign indicates that the digits that follow are in hexadecimal (base-16) format. The first two digits specify the alpha value, or opacity, of the color. FF indicates a color that is fully opaque, while 00 indicates a color that is completely transparent. The next six digits of the number specify the red, green, and blue values of the color.

The fill of the Ellipse is set to #FFFFFF00, which specifies a color that is fully opaque (FF), has the maximum amount of red (FF), the maximum amount of green (FF), and no blue(00). This combination produces yellow.

```
<Ellipse
  Fill="#FFFFFF00"
  CenterX="100"
  CenterY="200"
```

```
RadiusX="75"  
RadiusY="75"  
StrokeThickness="5"  
Stroke="#FF0000FF"/>
```

In the next example, the Fill and Stroke properties of an Ellipse are set using shorter hexadecimal notation. The alpha value is omitted, and one digit is used for each red, green, and blue value instead of two. The resulting Ellipse has colors identical to the first.

```
<Ellipse  
Fill="#FF0"  
CenterX="200"  
CenterY="200"  
RadiusX="75"  
RadiusY="75"  
StrokeThickness="5"  
Stroke="#00F"/>
```

In the next example, the Fill and Stroke properties of an Ellipse are set using two of the predefined colors, Yellow and Blue. The resulting Ellipse has colors identical to the previous two.

```
<Ellipse  
Fill="Yellow"  
CenterX="300"  
CenterY="200"  
RadiusX="75"  
RadiusY="75"  
StrokeThickness="5"  
Stroke="Blue"/>
```

In the final example, the Fill property of a Polyline is set by explicitly declaring a SolidColorBrush. The Color property of the SolidColorBrush is set to Blue, and its Opacity property is set to 0.4, creating a fill that is blue and 40 percent opaque (or 60 percent translucent).

```
<Polyline  
Points="300,200 400,125 400,275 300,200"  
Stroke="Purple"  
StrokeThickness="2.3">  
  
  <Polyline.Fill>  
    <SolidColorBrush Color="Blue" Opacity="0.4"/>  
  </Polyline.Fill>  
  
</Polyline>  
  
</Canvas>
```

This example demonstrates several equivalent ways to specify color values to fill an area using code. The simplest syntax uses a named color property from the Brushes class. The Colors class provides the same named color properties which you can pass as an argument to SolidColorBrush. The example also shows how to pass Color values to SolidColorBrush as separate alpha, red, green, and blue values using the Color structure's static FromScRGB method. The example draws identical Ellipse shapes using identical colors specified in these different ways.

```
// C#  
// Create the ellipses.  
Ellipse e1 = new Ellipse();  
Ellipse e2 = new Ellipse();
```

```

Ellipse e3 = new Ellipse();

// Set the fill value for the interior of each ellipse in
// different ways that have identical results.
e1.Fill = Brushes.Blue;
e2.Fill = new SolidColorBrush(Colors.Blue);
e3.Fill = new SolidColorBrush(Color.FromScRGB(1,0,0,1));

// Set the stroke value for the interior of each ellipse in
// different ways that have identical results.
e1.Stroke = Brushes.Black;
e2.Stroke = new SolidColorBrush(Colors.Black);
e3.Stroke = new SolidColorBrush(Color.FromScRGB(1,0,0,0));

```

This sample demonstrates only a few of the ways to instantiate Color objects. See Color for more information.

In order to actually render the ellipse the example also sets values for the StrokeThickness, CenterX, CenterY, RadiusX, and RadiusY properties in order to set the size and position of each ellipse.

```

// C#
// Set the thickness of the stroke.
e1.StrokeThickness = new Length(10);
e2.StrokeThickness = new Length(10);
e3.StrokeThickness = new Length(10);

// Set the size and position of the ellipses.
e1.CenterX = new Length(100);
e1.CenterY = new Length(75);
e1.RadiusX = new Length(50);
e1.RadiusY = new Length(50);

e2.CenterX = new Length(220);
e2.CenterY = new Length(75);
e2.RadiusX = new Length(50);
e2.RadiusY = new Length(50);

e3.CenterX = new Length(340);
e3.CenterY = new Length(75);
e3.RadiusX = new Length(50);
e3.RadiusY = new Length(50);

```

StartSegment Class

Definition: StartSegment

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this PathSegment. Inherited from

	PathSegment.
Copy	Creates a copy of this StartSegment.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DisableCore	Inherited from PathSegment.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from PathSegment.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this GradientStop that represents its current state. Inherited from PathSegment.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this BezierSegment that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from PathSegment.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the

SetDefaultParentTimelineCore	properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
StartSegment	Inherited from PathSegment.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the segment has animations.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
IsStroked	Gets or sets a Boolean that determines whether the segment is stroked. Inherited from PathSegment.
Point	
PointAnimations	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

SubLineCollection Class

Definition: collection of subline. Subline can be object of one of these types GlyphRun LineOver Inline object

Method	Description
CopyTo	copies the elements of the collection to an Array, starting at a particular Array index
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	Get enumerator for subline collection
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.

Property	Description
Count	Number of sublines in collection
IsSynchronized	Telling the caller if the collection is internally synchronized and thread-safe
SyncRoot	Collection's synchronization object

TileBrush Class

Definition: Abstract class that describes a way to fill a region with one or more "tiles." Derived classes define the different types of tiles that can be used; for example, the ImageBrush enables you to fill an area with an image.

Method	Description
CloneCore	Subclasses must implement this to provide clones of themselves. Inherited from Animatable.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this TileBrush.
Copy	Returns a modifiable copy of the brush. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Brush.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DeserializeFrom	Returns a new Brush initialized from the binary representation being read by the passed BinaryReader. Inherited from Brush.
DisableCore	Inherited from Brush.

EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Brush.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Brush.
GetCurrentValue	Returns the current value of the brush. The returned brush has the same value as the current object, but doesn't vary over time; that is, the returned brush is a snapshot of the current object at the point in time at which this method was called. Inherited from Brush.
GetCurrentValue	Returns a non-animated version of this TileBrush that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	Inherited from Brush.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Brush.
GetIsOverridingBaseValue	Inherited from Brush.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Brush.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Inherited from Brush.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SerializeOn	Serialize this object using the passed writer Inherited from Brush.
SetDefaultParentTimeline	Inherited from Brush.

SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	Inherited from Brush.
TileBrush	Initializes a new instance of the TileBrush class.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
ContentUnits	Gets or sets a BrushMappingMode enumeration that specifies whether the value of the brush's ViewBox—the size and position of the brush's content—is relative to the size of the output area. This property only has an effect when the size of the brush's ViewPort is set to Rect.Empty.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the brush has animations.
HorizontalAlignment	Gets or sets a HorizontalAlignment enumeration that specifies how the brush's content is horizontally aligned within its tiles.
IsAnimating	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
Opacity	Gets or sets the degree of opacity of a Brush. Inherited from Brush.
OpacityAnimations	Gets or sets the animations associated with the Opacity of the brush. Inherited from Brush.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Stretch	Gets or sets a Stretch enumeration that specifies how the brush's

	selected content (ViewBox) is displayed in the brush's tiles (ViewPort).
TileMode	Gets or sets a TileMode structure that specifies how the brush's tiles fill the output area.
Transform	Gets or sets a transformation that is applied to the brush. This transformation is applied after all other mapping and positioning have been processed. Inherited from Brush.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
VerticalAlignment	Gets or sets a VerticalAlignment enumeration that specifies how the brush's content is vertically aligned within its tiles.
ViewBox	Gets or sets the position and dimensions of the brush's content.
ViewBoxAnimations	Gets or sets a collection of RectModifier objects that animate the ViewBox of the brush.
ViewPort	Gets or sets the position and dimensions of the brush's tiles.
ViewPortAnimations	Gets or sets a collection of RectModifier objects that animate the ViewPort of the brush.
ViewPortUnits	Gets or sets a BrushMappingMode enumeration that specifies whether the value of the brush's ViewPort—the size and position of the brush's tiles—is relative to the size of the output area.

Transform Class

Definition: An abstract class that you use as the parent class of all types of transformations in a two-dimensional plane, including rotation (RotateTransform), scale (ScaleTransform), skew (SkewTransform), and translation (TranslateTransform). This class hierarchy differs from the Matrix structure both because it is a class and because it supports animation and enumeration semantics.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Subclasses must implement this to provide clones of themselves. Inherited from Animatable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this Transform.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane.
CreateTranslation	Creates a transformation used for translating in the x- and y-directions in a two-dimensional plane.

DisableCore	
Dispose	Dispose resources associated with transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base values set to the current animated values.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetHasAnimations	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	
GetIsOverridingBaseValue	
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimeline	
SetDefaultParentTimelineCore	

ToString	Not implemented. Use Object.ToString instead.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Returns true if any of the properties on this Changeable have animations attached. Inherited from Animatable.
Identity	Identity transformation.
IsAnimating	Returns true if any of the properties on this Changeable have animations attached that are currently animating their value. Inherited from Animatable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Returns true if any of the properties on this Changeable have animations attached that are currently affecting their value either by animating it or holding it in a fill state. Inherited from Animatable.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Value	Gets the current transformation as a Matrix object.

Use the MatrixTransform class to create custom transformations not provided by the RotateTransform, ScaleTransform, SkewTransform, and TranslateTransform classes.

A 3x3 matrix is used for transformations in a two-dimensional x-y plane. Affine transformation matrices can be multiplied to form any number of linear transformations, such as rotation and skew (shear), followed by translation. An affine transformation matrix has its final column equal to (0, 0, 1), so only the members in the first two columns need to be specified.

An "Avalon" Matrix has the following structure:

```
M11    M12    0
M21    M22    0
OffsetX OffsetY 1
```

The members in the last row, OffsetX and OffsetY, represent translation values. In methods and properties, the transformation matrix is usually specified as a vector with only six members, as follows:

(M11, M12, M21, M22, OffsetX, OffsetY)

This example specifies rotation transforms on several shapes. The Transform class enables you to reposition and apply rendering changes to shape elements such as Rectangle, Polyline, and Ellipse. The RotateTransform class specifies a rotation of the coordinate system. By default, the coordinate system is rotated about (0,0), the default origin, by r degrees. The following code provides some examples of different rotations, some in conjunction with translation.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml"
  xmlns:def="Definition">

  <!-- No transform -->
  <Polyline ID="box1"
    Stroke="Black"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10" />

  <!-- Rotate about a set origin, then translate -->
  <TransformDecorator AffectsLayout="false">
  <TransformDecorator.Transform>
    <TransformCollection>
      <RotateTransform Angle="90" />
      <TranslateTransform X="100" Y="100" />
    </TransformCollection>
  </TransformDecorator.Transform>
  <Polyline ID="box2"
    Stroke="Blue"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
  </Polyline>
  </TransformDecorator>

  <TransformDecorator AffectsLayout="false">
  <TransformDecorator.Transform>
    <TransformCollection>
      <RotateTransform Angle="-45" />
      <TranslateTransform X="100" Y="100" />
    </TransformCollection>
  </TransformDecorator.Transform>
  <Polyline ID="box3"
    Stroke="Orange"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
  </Polyline>
  </TransformDecorator>
```

```

<TransformDecorator AffectsLayout="false">
<TransformDecorator.Transform>
    <TransformCollection>
        <RotateTransform Angle="45" />
        <TranslateTransform X="100" Y="100" />
    </TransformCollection>
</TransformDecorator.Transform>
<Polyline ID="box4"
    Stroke="Green"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
</Polyline>
</TransformDecorator>

<!-- Translate, then rotate -->
<TransformDecorator AffectsLayout="false">
<TransformDecorator.Transform>
    <TransformCollection>
        <TranslateTransform X="200" Y="200" />
        <RotateTransform Angle="15" />
    </TransformCollection>
</TransformDecorator.Transform>
<Polyline ID="box5"
    Stroke="Cyan"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
</Polyline>
</TransformDecorator>
</Canvas>

```

When using rotation transformations, keep in mind that the transformation rotates the coordinate system for a particular element. This might apply effects that you should anticipate when working out the screen coordinates for elements. Depending on an element's position with respect to the origin, the effect of the rotation might not be to rotate it "in place." For an element positioned 200 units from 0 along the x-axis, for instance, a rotation of 30 degrees has the effect of swinging the element 30 degrees along a circle with radius 200, drawn around the origin.

The Center attribute on the RotateTransform element allows you to set an origin about which the shape will be rotated. Note that in the preceding example, the start points of the blue, orange, and green Polyline elements are all located at (100,100) because the Center attribute of those elements is (100,100). The cyan Polyline, however, was rotated 15 degrees on an arc from (200,200) because its TranslateTransform was applied first.

This example uses a RotateTransform object to rotate a Shape element. To scale an element, use a ScaleTransform object to specify the transformation, and a TransformDecorator object to apply the transformation. Instead of adding the element to the Canvas element or other Panel element, add the element as child of the TransformDecorator and add the TransformDecorator to the Panel. The RotateTransform object specifies a rotation of the coordinate system. By default, the coordinate system is rotated about (0,0), the default origin, by r degrees. The following code shows the two shapes. The first shape is not rotated. The second shape is rotated by 45 degrees around a RotateTransform of (110,110), the lower right corner of the shape. As a result, the shape appears as though it was rotated about its corner.

```

// C#
// Create a canvas to contain the shapes.
Canvas myCanvas = new Canvas();

```

```

// Create a PointCollection to contain the points of the Polygon shapes.
PointCollection myPointCollection = new PointCollection();
myPointCollection.Add(new Point(60, 60));
myPointCollection.Add(new Point(70, 70));
myPointCollection.Add(new Point(70, 110));
myPointCollection.Add(new Point(110, 110));
myPointCollection.Add(new Point(110, 70));
myPointCollection.Add(new Point(70, 70));

// Create the first Polygon and add it to the Canvas.
Polygon box1 = new Polygon();
box1.Points = myPointCollection;
box1.Stroke = Brushes.Black;
box1.StrokeThickness = new Length(5);
myCanvas.Children.Add(box1);

// Create the second Polygon. This polygon contains the same points as the
// first, but is rotated.
Polygon box2 = new Polygon();
box2.Points = myPointCollection;
box2.Stroke = Brushes.Blue;
box2.StrokeThickness = new Length(5);

// Create a TransformDecorator to transform the shape.
TransformDecorator transformer = new TransformDecorator();
transformer.AffectsLayout = false;

// Create the rotation transformation.
RotateTransform myRotateTransform = new RotateTransform(45, new Point(110,110));

// Set the TransformDecorator.Transform property.
transformer.Transform = myRotateTransform;

// Add box2 as a child of the TransformDecorator and add the TransformDecorator
// to the Canvas.
transformer.Child = box2;
myCanvas.Children.Add(transformer);

```

```

' VB .NET
' Create a PointCollection to contain the points of the
' Polygon shapes.
Dim myPointCollection As MSAvalon.Windows.Media.PointCollection
myPointCollection = new MSAvalon.Windows.Media.PointCollection
myPointCollection.Add(new MSAvalon.Windows.Point(60, 60))
myPointCollection.Add(new MSAvalon.Windows.Point(70, 70))
myPointCollection.Add(new MSAvalon.Windows.Point(70, 110))
myPointCollection.Add(new MSAvalon.Windows.Point(110, 110))
myPointCollection.Add(new MSAvalon.Windows.Point(110, 70))
myPointCollection.Add(new MSAvalon.Windows.Point(70, 70))

' Create the first Polygon and add it to the Canvas.
Dim box1 As MSAvalon.Windows.Shapes.Polygon
box1 = new MSAvalon.Windows.Shapes.Polygon
box1.Points = myPointCollection

```

```

box1.Stroke = Brushes.Black
box1.StrokeThickness = new MS Avalon.Windows.Length(5)
myCanvas.Children.Add(box1)

' Create the second Polygon. This polygon contains the same
' points as the first, but is rotated.
Dim box2 As MS Avalon.Windows.Shapes.Polygon
box2 = new MS Avalon.Windows.Shapes.Polygon
box2.Points = myPointCollection
box2.Stroke = Brushes.Blue
box2.StrokeThickness = new MS Avalon.Windows.Length(5)

' Create a TransformDecorator to transform the shape.
Dim transformer As MS Avalon.Windows.Controls.TransformDecorator
transformer = new MS Avalon.Windows.Controls.TransformDecorator
transformer.AffectsLayout = false

' Create the scale transformation.
Dim myRotateTransform As _
    new MS Avalon.Windows.Media.RotateTransform(45, _
    new MS Avalon.Windows.Point(110,110))

' Set the TransformDecorator.Transform property
transformer.Transform = myRotateTransform

' Add box2 as a child of the TransformDecorator and add
' the TransformDecorator to the Canvas.
transformer.Child = box2
myCanvas.Children.Add(transformer)

```

When using rotation transformations, keep in mind that the transformation rotates the coordinate system for a particular element. This might apply effects that you should anticipate when working out the screen coordinates for elements. Depending on an element's position with respect to the origin, the effect of the rotation might not be to rotate it "in place." For example, for an element positioned 200 units from 0 along the x-axis a rotation of 30 degrees has the effect of swinging the element 30 degrees along a circle with radius 200, drawn around the origin.

Although the previous example shows how to rotate a Shape element, the process is the same for other elements, including Button and Panel elements.

This example specifies scale transformations on several shapes. The ScaleTransform class enables you to reposition and apply rendering changes to shape elements like Rectangle, Polygon, and Ellipse.

The ScaleX and ScaleY properties resize an element according to the factor you specify. For example, setting ScaleX to 1.5 resizes the element's X-axis value at 150 percent.

The following example demonstrates a polygon scaled in several ways. The first polygon (black) reflects no transformation. The second polygon (blue) is scaled at 50 percent in Y, and the third polygon (red) is scaled at 50 percent in X. Two of the polygons also include translation transformations for repositioning. Note that the scale transformation affects the element's stroke-thickness as well as its size.

```

<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <!-- No transform specified -->
  <Polygon ID="star1"
    Stroke="black"
    StrokeThickness="2.0"
    Points="176.5,50 189.2,155.003 286.485,113.5 201.9,177 286.485,240.5

```

```

189.2,198.997 176.5,304 163.8,198.997 66.5148,240.5 151.1,177 66.5148,113.5
163.8,155.003">
</Polygon>

<!-- Scaled in Y -->
<TransformDecorator AffectsLayout="false">
  <TransformDecorator.Transform>
    <TransformCollection>
      <ScaleTransform ScaleX="1" ScaleY="0.5" />
      <TranslateTransform X="100" Y="0" />
    </TransformCollection>
  </TransformDecorator.Transform>
  <Polygon ID="star2"
  Stroke="blue"
  StrokeThickness="2.0"
  Points="176.5,50 189.2,155.003 286.485,113.5 201.9,177 286.485,240.5
  189.2,198.997 176.5,304 163.8,198.997 66.5148,240.5 151.1,177 66.5148,113.5
  163.8,155.003" />
</TransformDecorator>

<!-- Scaled in X -->
<TransformDecorator AffectsLayout="false">
  <TransformDecorator.Transform>
    <TransformCollection>
      <ScaleTransform ScaleX="0.5" ScaleY="1" />
      <TranslateTransform X="0" Y="100" />
    </TransformCollection>
  </TransformDecorator.Transform>
  <Polygon ID="star3"
  Stroke="red"
  StrokeThickness="2.0"
  Points="176.5,50 189.2,155.003 286.485,113.5 201.9,177 286.485,240.5
  189.2,198.997 176.5,304 163.8,198.997 66.5148,240.5 151.1,177 66.5148,113.5
  163.8,155.003" />
</TransformDecorator>

</Canvas>

```

This example specifies scale transformations on several shapes. To transform an element, use a `TransformDecorator` and set its `Transform` to the transformation you wish to apply. Set the element to transform as the `TransformDecorator` object's `Child`, and add the `TransformDecorator` to a panel. The `ScaleTransform` class, one of the transformations you can use with a `TransformDecorator`, enables you to reposition and apply rendering changes elements like `Rectangle`, `Polygon`, and `Button`.

The `ScaleX` and `ScaleY` properties resize an element according to the factor you specify. For example, setting `ScaleX` to 1.5 resizes the element's x-axis value at 150 percent.

The following example demonstrates a polygon scaled in several ways. The first polygon (black) reflects no transformation.

```

// #C
// Create a canvas to contain the shapes.
Canvas myCanvas = new Canvas();

// Create a PointCollection to contain the points of the
// Polygon shapes.

```

```

PointCollection myPointCollection = new PointCollection();
myPointCollection.Add(new Point(176.5, 50));
myPointCollection.Add(new Point(189, 155));
myPointCollection.Add(new Point(286, 113));
myPointCollection.Add(new Point(201, 177));
myPointCollection.Add(new Point(286, 240));
myPointCollection.Add(new Point(189, 198));
myPointCollection.Add(new Point(176, 304));
myPointCollection.Add(new Point(163, 198));
myPointCollection.Add(new Point(66, 240));
myPointCollection.Add(new Point(151, 177));
myPointCollection.Add(new Point(66, 113));
myPointCollection.Add(new Point(163, 155));

```

```

// Create the first Polygon.
Polygon firstPolygon = new Polygon();
firstPolygon.Points = myPointCollection;
firstPolygon.Stroke = Brushes.Black;
firstPolygon.StrokeThickness = new Length(2);
myCanvas.Children.Add(firstPolygon);

```

```

' VB .NET
' Create a PointCollection to contain the points of the
' Polygon shapes.
Dim myPointCollection As new MS Avalon.Windows.Media.PointCollection
myPointCollection.Add(new MS Avalon.Windows.Point(176.5, 50))
myPointCollection.Add(new MS Avalon.Windows.Point(189, 155))
myPointCollection.Add(new MS Avalon.Windows.Point(286, 113))
myPointCollection.Add(new MS Avalon.Windows.Point(201, 177))
myPointCollection.Add(new MS Avalon.Windows.Point(286, 240))
myPointCollection.Add(new MS Avalon.Windows.Point(189, 198))
myPointCollection.Add(new MS Avalon.Windows.Point(176, 304))
myPointCollection.Add(new MS Avalon.Windows.Point(163, 198))
myPointCollection.Add(new MS Avalon.Windows.Point(66, 240))
myPointCollection.Add(new MS Avalon.Windows.Point(151, 177))
myPointCollection.Add(new MS Avalon.Windows.Point(66, 113))
myPointCollection.Add(new MS Avalon.Windows.Point(163, 155))

```

```

' Create the first Polygon.
Dim firstPolygon As new MS Avalon.Windows.Shapes.Polygon
firstPolygon.Points = myPointCollection
firstPolygon.Stroke = MS Avalon.Windows.Media.Brushes.Black
firstPolygon.StrokeThickness = new MS Avalon.Windows.Length(2)
myCanvas.Children.Add(firstPolygon)

```

The second polygon (blue) is scaled at 50 percent in y and translated by 150 pixels. Because there are multiple transformations, a TransformCollection object is used to contain the transformations and set the value of the TransformDecorator. Note that the scale transformation affects the element's stroke thickness as well as its size. In order to for the TranslateTransform to render, the AffectsLayout property of the TransformDecorator must be set to false.

```

// C#
// Create the second Polygon. This polygon contains the same
// points as the first, but is scaled and translated.

```

```

Polygon secondPolygon = new Polygon();
secondPolygon.Points = myPointCollection;
secondPolygon.Stroke = Brushes.Blue;
secondPolygon.StrokeThickness = new Length(2);

// Add a semi-transparent gradient fill to make the shape stand out.
RadialGradientBrush myGradient =
    new RadialGradientBrush(Colors.Blue, Colors.LimeGreen);
myGradient.Opacity = 0.4;
secondPolygon.Fill = myGradient;

// Create a TransformDecorator to transform secondPolygon.
TransformDecorator transformer = new TransformDecorator();
transformer.AffectsLayout = false;

// Create the scale and translate transformations.
ScaleTransform myScaleTransform = new ScaleTransform(1, 0.5);
TranslateTransform myTranslateTransform = new TranslateTransform(150, 0);

//Create a collection to contain the transformations.
TransformCollection transformations = new TransformCollection();

transformations.Add(myScaleTransform);
transformations.Add(myTranslateTransform);
transformer.Transform = transformations;

// Add secondPolygon as a child of the TransformDecorator.
transformer.Child = secondPolygon;

// Add the decorator to the Canvas.
myCanvas.Children.Add(transformer);

```

```

' VB .NET
' Create the second Polygon. This polygon contains the same
' points as the first, but is scaled and translated.
Dim secondPolygon As new MS Avalon.Windows.Shapes.Polygon
secondPolygon.Points = myPointCollection
secondPolygon.Stroke = MS Avalon.Windows.Media.Brushes.Blue
secondPolygon.StrokeThickness = new MS Avalon.Windows.Length(2)

' Add a semi-transparent gradient fill to make the shape stand out.
Dim myGradient As new MS Avalon.Windows.Media.RadialGradientBrush( _
    Colors.Blue, Colors.LimeGreen)
myGradient.Opacity = 0.4
secondPolygon.Fill = myGradient

' Create a TransformDecorator to transform secondPolygon.
Dim transformer As new MS Avalon.Windows.Controls.TransformDecorator
transformer.AffectsLayout = false

' Create the scale and translate transformations.
Dim myScaleTransform As new MS Avalon.Windows.Media.ScaleTransform(1, 0.5)
Dim myTranslateTransform As new MS Avalon.Windows.Media.TranslateTransform(150, 0)

' Create a collection to contain the transformations.

```

Dim transformations As new MS Avalon.Windows.Media.TransformCollection()

```
transformations.Add(myScaleTransform)
transformations.Add(myTranslateTransform)
transformer.Transform = transformations
```

' Add secondPolygon as a child of the TransformDecorator.
transformer.Child = secondPolygon

' Add the decorator to the Canvas.
myCanvas.Children.Add(transformer)

This example uses classes from the MS Avalon.Windows, MS Avalon.Windows.Media, and MS Avalon.Windows.Controls namespaces.

This example specifies translation transforms on several shapes. The Transform class enables you to reposition and apply rendering changes to shape elements like Rectangle, Polygon, and Ellipse.

The translate value setting repositions the current coordinate system, locating the new origin of that coordinate system at the specified values. In the following example, two Polyline shapes indicate the original coordinate system and a translation to (50, 50).

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <!-- No translation -->
  <Polyline ID="angle1"
    Stroke="red"
    StrokeThickness="10"
    Points="10,160 10,10 160,10">
  </Polyline>

  <!-- Translation in x- and y- axes. -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection >
        <TranslateTransform X="50" Y="50" />
      </TransformCollection>
    </TransformDecorator.Transform>

    <Polyline ID="angle2"
      Stroke="blue"
      StrokeThickness="10"
      Points="10,160 10,10 160,10">
    </Polyline>

  </TransformDecorator>

</Canvas>
```

When using translations with other transformations, consider how the translation will affect the computation of those transformations. For example, consider a shape positioned at (100, 100). If the coordinate system is rotated by thirty degrees, the shape occupies a different screen position than if it had been positioned at (0, 0) when the rotation was computed.


```

<DockPanel ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

    <!-- Shape with no transformation -->
    <Polyline ID="box1"
    Stroke="black"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10" />

    <!-- Translation only -->
    <TransformDecorator AffectsLayout="false">
        <TransformDecorator.Transform>
            <TransformCollection >
                <TranslateTransform X="100" Y="100" />
            </TransformCollection>
        </TransformDecorator.Transform>
    </TransformDecorator>
    <Polyline ID="box2"
    Stroke="red"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
        </Polyline>
    </TransformDecorator>

    <!-- Rotate at origin, then translate -->
    <TransformDecorator AffectsLayout="false">
        <TransformDecorator.Transform>
            <TransformCollection >
                <RotateTransform Center="0 0" Angle="-30" />
                <TranslateTransform X="100" Y="100" />
            </TransformCollection>
        </TransformDecorator.Transform>
    </TransformDecorator>
    <Polyline ID="box3"
    Stroke="blue"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
        </Polyline>
    </TransformDecorator>

    <!-- Translate, then rotate the new coordinate system (note the different
    element position) -->
    <TransformDecorator AffectsLayout="false">
        <TransformDecorator.Transform>
            <TransformCollection >
                <TranslateTransform X="100" Y="100" />
                <RotateTransform Center="0 0" Angle="-30" />
            </TransformCollection>
        </TransformDecorator.Transform>
    </TransformDecorator>
    <Polyline ID="box4"
    Stroke="orange"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
        </Polyline>
    </TransformDecorator>

</DockPanel>

```

For many rotation, scale, and skew transformations, it can be helpful to translate a positioned element to the origin, apply the rotation, scale, or skew value, and then translate the element back to its intended screen position. This practice makes it less cumbersome to account for specific element positions across transformations because transformed elements reflect specific positioning values explicitly set in markup.

This example demonstrates how to use the Transform property of brushes to apply transformations to LinearGradientBrush and RadialGradientBrush fills. A LinearGradientBrush is used to fill the first two Rectangle elements. The difference between the rectangles is that the LinearGradientBrush in the second rectangle is rotated 45 degrees. The second pair of rectangles illustrates the before and after effect of a ScaleTransform by reducing a RadialGradientBrush to half its normal height.

```
<Border xmlns="http://schemas.microsoft.com/2003/xaml"
Background="#CCCCCC">

<Canvas Height="40">

<!-- Rectangle #1 is filled with a LinearGradientBrush. The gradient colors
flow from left to right by default. -->

<Rectangle RectangleLeft="10" RectangleTop="10"
RectangleWidth="300" RectangleHeight="200">

<Rectangle.Fill>
<LinearGradientBrush>

<LinearGradientBrush.GradientStops>
<GradientStopCollection>
<GradientStop Color="red" Offset="0"/>
<GradientStop Color="yellow" Offset="1" />
<GradientStop Color="blue" Offset="0.5"/>
<GradientStop Color="white" Offset="0.2"/>
</GradientStopCollection>
</LinearGradientBrush.GradientStops>

</LinearGradientBrush>
</Rectangle.Fill>

</Rectangle>

<!-- Rectangle #2 is identical to the first rectangle except that the Transform
property rotates the LinearGradientBrush so that the gradient colors are
rotated by 45 degrees. -->

<Rectangle RectangleLeft="320" RectangleTop="10"
RectangleWidth="300" RectangleHeight="200">

<Rectangle.Fill>
<LinearGradientBrush>

<LinearGradientBrush.Transform>
<RotateTransform Angle="45" /> <!-- Rotation angle. -->
</LinearGradientBrush.Transform>

<LinearGradientBrush.GradientStops>
<GradientStopCollection>
```

```

        <GradientStop Color="red" Offset="0"/>
        <GradientStop Color="yellow" Offset="1" />
        <GradientStop Color="blue" Offset="0.5"/>
        <GradientStop Color="white" Offset="0.2"/>
    </GradientStopCollection>
</LinearGradientBrush.GradientStops>

</LinearGradientBrush>
</Rectangle.Fill>

</Rectangle>

<!-- Rectangle #3 is filled with a RadialGradientBrush. -->

<Rectangle RectangleLeft="10" RectangleTop="250"
RectangleWidth="300" RectangleHeight="200">

    <Rectangle.Fill>
        <RadialGradientBrush Focus="0.5,0.5">

            <RadialGradientBrush.GradientStops>
                <GradientStopCollection>
                    <GradientStop Color="red" Offset="0"/>
                    <GradientStop Color="yellow" Offset="1"/>
                    <GradientStop Color="blue" Offset="0.5"/>
                </GradientStopCollection>
            </RadialGradientBrush.GradientStops>

        </RadialGradientBrush>
    </Rectangle.Fill>

</Rectangle>

<!-- Rectangle #4 is identical to the third rectangle except that the Transform
property applies a ScaleTransform to the RadialGradientBrush so that the
gradient is half its previous height. -->

<Rectangle RectangleLeft="320" RectangleTop="250"
RectangleWidth="300" RectangleHeight="200">

    <Rectangle.Fill>

        <RadialGradientBrush Focus="0.5,0.5">
            <RadialGradientBrush.Transform>
                <ScaleTransform ScaleX="1" ScaleY="0.5" /><!-- Scale transform. -->
            </RadialGradientBrush.Transform>

            <RadialGradientBrush.GradientStops>
                <GradientStopCollection>
                    <GradientStop Color="red" Offset="0"/>
                    <GradientStop Color="yellow" Offset="1"/>
                    <GradientStop Color="blue" Offset="0.5"/>
                </GradientStopCollection>
            </RadialGradientBrush.GradientStops>
        </RadialGradientBrush>
    </Rectangle.Fill>
</Rectangle>

```

```

        </RadialGradientBrush>
    </Rectangle.Fill>

</Rectangle>

</Canvas>
</Border>

```

This example shows how to apply a ScaleTransform when an event occurs. The concept is the same for applying other types of transformations. See the Transform class for more information about the available types of transformations.

To apply a specific transformation to an element, you must create a TransformDecorator that you will add to that element. In markup, wrap the element with the TransformDecorator. When the event occurs, set the TransformProperty property of the TransformDecorator to the appropriate transformation, in this case a ScaleTransform.

```

<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml/"
xmlns:def="Definition">

<TransformDecorator ID="td1" AffectsLayout="false">
  <Button ID="Button1" MouseEnter="Enter" MouseLeave="Leave">Button</Button>
</TransformDecorator>

<def:Code>

<![CDATA[
    Private Sub Enter(ByVal sender as object, ByVal args as MS Avalon.Windows.Input.MouseEventArgs)
        Dim scaler As new ScaleTransform(2,2)
        td1.Transform = scaler
    End Sub

    Private Sub Leave(ByVal sender as object, ByVal e as MS Avalon.Windows.Input.MouseEventArgs)
        Dim scaler As new ScaleTransform(1,1)
        td1.Transform = scaler
    End Sub

]]>

</def:Code>

</Canvas>

```

TransformCollection Class

Definition: Used to create and manipulate a list of Transform objects.

Method	Description
Add	Adds a Transform object to the end of the collection.
AddMatrix	Adds a Matrix or a set of matrix values to the collection and returns the index position at which the object was added.
AddRange	Adds the transformations contained in the passed TransformCollection to the current collection.

AddRotate	Constructs and adds a RotateTransform to the list and returns the location at which the transformation was added.
AddScale	Constructs and adds a ScaleTransform to the list and returns the location at which the transformation was added.
AddSkew	Constructs and adds a SkewTransform to the list and returns the location at which the transformation was added.
AddTranslate	Constructs and adds a TranslateTransform to the list and returns the location at which the transformation was added.
Clear	Removes all transformations from the collection.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Implementation of Animatable.CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a Boolean that indicates whether the specified transformation is contained within the collection.
Copy	Creates a copy of this Transform. Inherited from Transform.
Copy	Creates a copy of this TransformCollection.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	Copies the entire TransformCollection to a compatible one-dimensional Array, starting at the specified index of the target array.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane. Inherited from Transform.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane. Inherited from Transform.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane. Inherited from Transform.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane. Inherited from Transform.
CreateTranslation	Creates a transformation used for translating in the x- and y-directions in a two-dimensional plane. Inherited from Transform.
DisableCore	
Dispose	Dispose resources associated with transform. Inherited from Transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Transform.
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base

	values set to the current animated values. Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Returns a non-animated version of this TransformCollection that represents its current state.
GetEnumerator	Returns an enumerator that can iterate through the TransformCollection.
GetHasAnimations	Inherited from Transform.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Transform.
GetIsOverridingBaseValue	Inherited from Transform.
GetOptimizedTransform	Collapse transformation list to a base transformation.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	Returns the index of the specified Transform if it exists in the collection or part of the collection. If the Transform isn't in the collection, this method returns -1.
Insert	Inserts a Transform object into the collection at the specified index.
InsertRange	Inserts the Transform objects contained within the specified TransformCollection into the collection at the specified zero-based index.
LastIndexOf	Searches for the specified Transform and returns the zero-based index of the last occurrence within all or part of the TransformCollection.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	

ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	Removes the specified Transform object from the collection, if it exists.
RemoveAt	Removes the Transform object at the specified index from the collection.
RemoveRange	Removes the specified number of Transform objects from the section of the collection that starts at the specified index.
SetDefaultParentTimeline	Inherited from Transform.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimelineCore	
SetRange	Copies the Transform objects of a TransformCollection over a range of objects in the current collection.
ToString	Not implemented. Use Object.ToString instead. Inherited from Transform.
TransformCollection	Initializes a new instance of the TransformCollection class.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	Gets or sets the number of items the collection can store.
Count	Gets the number of transformations contained in the Count.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether any of the transformations in the collection have animations.
Identity	Identity transformation. Inherited from Transform.
IsAnimating	Returns true if the list contains transforms that contain active

	animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
IsOverridingBaseValue	
Item	Gets or sets the transformation at the specified index in the transformation list.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Value	Gets the current transformation list as a Matrix object.

This example specifies scale transformations on several shapes. To transform an element, use a TransformDecorator and set its Transform to the transformation you wish to apply. Set the element to transform as the TransformDecorator object's Child, and add the TransformDecorator to a panel. The ScaleTransform class, one of the transformations you can use with a TransformDecorator, enables you to reposition and apply rendering changes elements like Rectangle, Polygon, and Button.

The ScaleX and ScaleY properties resize an element according to the factor you specify. For example, setting ScaleX to 1.5 resizes the element's x-axis value at 150 percent.

The following example demonstrates a polygon scaled in several ways. The first polygon (black) reflects no transformation.

```
// #C
// Create a canvas to contain the shapes.
Canvas myCanvas = new Canvas();

// Create a PointCollection to contain the points of the
// Polygon shapes.
PointCollection myPointCollection = new PointCollection();
myPointCollection.Add(new Point(176.5, 50));
myPointCollection.Add(new Point(189, 155));
myPointCollection.Add(new Point(286, 113));
myPointCollection.Add(new Point(201, 177));
myPointCollection.Add(new Point(286, 240));
myPointCollection.Add(new Point(189, 198));
myPointCollection.Add(new Point(176, 304));
myPointCollection.Add(new Point(163, 198));
myPointCollection.Add(new Point(66, 240));
myPointCollection.Add(new Point(151, 177));
myPointCollection.Add(new Point(66, 113));
myPointCollection.Add(new Point(163, 155));

// Create the first Polygon.
Polygon firstPolygon = new Polygon();
firstPolygon.Points = myPointCollection;
firstPolygon.Stroke = Brushes.Black;
firstPolygon.StrokeThickness = new Length(2);
```



```
myCanvas.Children.Add(firstPolygon);
```

```
' VB .NET
```

```
' Create a PointCollection to contain the points of the
```

```
' Polygon shapes.
```

```
Dim myPointCollection As new MS Avalon.Windows.Media.PointCollection
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(176.5, 50))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(189, 155))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(286, 113))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(201, 177))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(286, 240))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(189, 198))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(176, 304))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(163, 198))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(66, 240))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(151, 177))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(66, 113))
```

```
myPointCollection.Add(new MS Avalon.Windows.Point(163, 155))
```

```
' Create the first Polygon.
```

```
Dim firstPolygon As new MS Avalon.Windows.Shapes.Polygon
```

```
firstPolygon.Points = myPointCollection
```

```
firstPolygon.Stroke = MS Avalon.Windows.Media.Brushes.Black
```

```
firstPolygon.StrokeThickness = new MS Avalon.Windows.Length(2)
```

```
myCanvas.Children.Add(firstPolygon)
```

The second polygon (blue) is scaled at 50 percent in y and translated by 150 pixels. Because there are multiple transformations, a TransformCollection object is used to contain the transformations and set the value of the TransformDecorator. Note that the scale transformation affects the element's stroke thickness as well as its size. In order to for the TranslateTransform to render, the AffectsLayout property of the TransformDecorator must be set to false.

```
// C#
```

```
// Create the second Polygon. This polygon contains the same
```

```
// points as the first, but is scaled and translated.
```

```
Polygon secondPolygon = new Polygon();
```

```
secondPolygon.Points = myPointCollection;
```

```
secondPolygon.Stroke = Brushes.Blue;
```

```
secondPolygon.StrokeThickness = new Length(2);
```

```
// Add a semi-transparent gradient fill to make the shape stand out.
```

```
RadialGradientBrush myGradient =
```

```
    new RadialGradientBrush(Colors.Blue, Colors.LimeGreen);
```

```
myGradient.Opacity = 0.4;
```

```
secondPolygon.Fill = myGradient;
```

```
// Create a TransformDecorator to transform secondPolygon.
```

```
TransformDecorator transformer = new TransformDecorator();
```

```
transformer.AffectsLayout = false;
```

```
// Create the scale and translate transformations.
```

```
ScaleTransform myScaleTransform = new ScaleTransform(1, 0.5);
```

```
TranslateTransform myTranslateTransform = new TranslateTransform(150, 0);
```

```

//Create a collection to contain the transformations.
TransformCollection transformations = new TransformCollection();

transformations.Add(myScaleTransform);
transformations.Add(myTranslateTransform);
transformer.Transform = transformations;

// Add secondPolygon as a child of the TransformDecorator.
transformer.Child = secondPolygon;

// Add the decorator to the Canvas.
myCanvas.Children.Add(transformer);

' VB .NET
' Create the second Polygon. This polygon contains the same
' points as the first, but is scaled and translated.
Dim secondPolygon As new MS Avalon.Windows.Shapes.Polygon
secondPolygon.Points = myPointCollection
secondPolygon.Stroke = MS Avalon.Windows.Media.Brushes.Blue
secondPolygon.StrokeThickness = new MS Avalon.Windows.Length(2)

' Add a semi-transparent gradient fill to make the shape stand out.
Dim myGradient As new MS Avalon.Windows.Media.RadialGradientBrush( _
    Colors.Blue, Colors.LimeGreen)
myGradient.Opacity = 0.4
secondPolygon.Fill = myGradient

' Create a TransformDecorator to transform secondPolygon.
Dim transformer As new MS Avalon.Windows.Controls.TransformDecorator
transformer.AffectsLayout = false

' Create the scale and translate transformations.
Dim myScaleTransform As new MS Avalon.Windows.Media.ScaleTransform(1, 0.5)
Dim myTranslateTransform As new MS Avalon.Windows.Media.TranslateTransform(150, 0)

' Create a collection to contain the transformations.
Dim transformations As new MS Avalon.Windows.Media.TransformCollection()

transformations.Add(myScaleTransform)
transformations.Add(myTranslateTransform)
transformer.Transform = transformations

' Add secondPolygon as a child of the TransformDecorator.
transformer.Child = secondPolygon

' Add the decorator to the Canvas.
myCanvas.Children.Add(transformer)

```

TransformConverter Class

Definition: Used to convert a Transform object to or from another object type.

Method	Description
CanConvertFrom	Determines whether this class can convert an object of a given type to a

	Transform type. Currently only string objects can be converted.
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	Determines whether this class can convert an object of a given type to the specified destination type.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	Converts from an object of a given type to a Transform object.
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	Converts the given value object to the specified type, using the specified context and culture information.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.
TransformConverter	Initializes a new instance of the TransformConverter class as a TypeConverter object.

TranslateTransform Class

Definition: Translates an object in the two-dimensional x-y plane.

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes

CloneCore	that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Implementation of Animatable.CloneCore.
Copy	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this TranslateTransform.
Copy	Creates a copy of this Transform. Inherited from Transform.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CreateMatrix	Creates an arbitrary affine transformation matrix used for transformations in a two-dimensional plane. Inherited from Transform.
CreateRotation	Creates a transformation used for rotation in a two-dimensional plane. Inherited from Transform.
CreateScale	Creates a transformation used for scaling in a two-dimensional plane. Inherited from Transform.
CreateSkew	Creates a transformation used for skewing in a two-dimensional plane. Inherited from Transform.
CreateTranslation	Creates a transformation used for translating in the x- and y- directions in a two-dimensional plane. Inherited from Transform.
DisableCore	Inherited from Transform.
Dispose	Dispose resources associated with transform. Inherited from Transform.
EmbeddedAnimationCollectionReader	Maybe this should be internal?? Inherited from Animatable.
EmbeddedAnimationCollectionWriter	Inherited from Animatable.
EmbeddedChangeableReader	Inherited from Animatable.
EmbeddedChangeableWriter	Inherited from Animatable.
EnableCore	Inherited from Transform.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Inherited from Transform.
GetCurrentValue	Returns a non-animated version of this TranslateTransform that represents its current state.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state. Inherited from Animatable.
GetCurrentValue	Creates a non-animated copy of this Transform with all of its base values set to the current animated values. Inherited from Transform.
GetHasAnimations	Inherited from Transform.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetIsAnimating	Inherited from Transform.
GetIsOverridingBaseValue	Inherited from Transform.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.

MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from Transform.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent. Inherited from Animatable.
SetDefaultParentTimeline	Inherited from Transform.
SetDefaultParentTimelineCore	Inherited from Transform.
ToString	Not implemented. Use Object.ToString instead. Inherited from Transform.
TranslateTransform	Initializes a new instance of the TranslateTransform class.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method. Inherited from Animatable.
HasAnimations	Gets a Boolean that indicates whether the transformation has

	animations.
Identity	Identity transformation. Inherited from Transform.
IsAnimating	Returns true if the transformation contains active animations.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Value	Gets the current translation transformation as a Matrix object.
X	Gets or sets the x-direction component of the translation.
XAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's x-axis displacement.
Y	Gets or sets the y-direction component of the translation.
YAnimations	Gets or sets a collection of DoubleModifier objects that animate the transformation's y-axis displacement.

TranslateTransform defines an axis-aligned translation in the x and y directions. The static matrix representation for a translation by offset (dx, dy) is as follows:

To translate a Shape element, use a TransformDecorator.

This example specifies translation transforms on several shapes. The Transform class enables you to reposition and apply rendering changes to shape elements like Rectangle, Polygon, and Ellipse. The translate value setting repositions the current coordinate system, locating the new origin of that coordinate system at the specified values. In the following example, two Polyline shapes indicate the original coordinate system and a translation to (50, 50).

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <!-- No translation -->
  <Polyline ID="angle1"
    Stroke="red"
    StrokeThickness="10"
    Points="10,160 10,10 160,10">
  </Polyline>

  <!-- Translation in x- and y- axes. -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection >
        <TranslateTransform X="50" Y="50" />
      </TransformCollection>
    </TransformDecorator.Transform>

    <Polyline ID="angle2"
      Stroke="blue"
```

```

StrokeThickness="10"
Points="10,160 10,10 160,10">
</Polyline>

```

```
</TransformDecorator>
```

```
</Canvas>
```

When using translations with other transformations, consider how the translation will affect the computation of those transformations. For example, consider a shape positioned at (100, 100). If the coordinate system is rotated by thirty degrees, the shape occupies a different screen position than if it had been positioned at (0, 0) when the rotation was computed.

```

<DockPanel ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

  <!-- Shape with no transformation -->
  <Polyline ID="box1"
    Stroke="black"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10" />

  <!-- Translation only -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection >
        <TranslateTransform X="100" Y="100" />
      </TransformCollection>
    </TransformDecorator.Transform>
  <Polyline ID="box2"
    Stroke="red"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
    </Polyline>
  </TransformDecorator>

  <!-- Rotate at origin, then translate -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection >
        <RotateTransform Center="0 0" Angle="-30" />
        <TranslateTransform X="100" Y="100" />
      </TransformCollection>
    </TransformDecorator.Transform>
  <Polyline ID="box3"
    Stroke="blue"
    StrokeThickness="5"
    Points="0,0 10,10 10,50 50,50 50,10 10,10">
    </Polyline>
  </TransformDecorator>

  <!-- Translate, then rotate the new coordinate system (note the different
element position) -->
  <TransformDecorator AffectsLayout="false">
    <TransformDecorator.Transform>
      <TransformCollection >

```

```

        <TranslateTransform X="100" Y="100" />
        <RotateTransform Center="0 0" Angle="-30" />
    </TransformCollection>
</TransformDecorator.Transform>
<Polyline ID="box4"
Stroke="orange"
StrokeThickness="5"
Points="0,0 10,10 10,50 50,50 50,10 10,10">
    </Polyline>
</TransformDecorator>

</DockPanel>

```

For many rotation, scale, and skew transformations, it can be helpful to translate a positioned element to the origin, apply the rotation, scale, or skew value, and then translate the element back to its intended screen position. This practice makes it less cumbersome to account for specific element positions across transformations because transformed elements reflect specific positioning values explicitly set in markup.

Typeface Class

Definition: A Typeface is a combination of family, weight, style and stretch.

Method	Description
Equals	Equality check
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetAdaptiveLayoutMetrics	Get font BlackRiver metrics for adaptive layout process
GetHashCode	Create correspondent hash code for the object
GetType	Gets the Type of the current instance. Inherited from Object.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
Typeface	Construct a typeface

Property	Description
CapsHeight	Distance from baseline to top of English capital, relative to em size.
FontFamily	Font family
Stretch	Font Stretch (narrow, wide, etc.)
StrikeoutPosition	Distance from baseline to strike-through position
StrikeoutThickness	strike-through thickness
Style	Font style (italic, oblique)
UnderlinePosition	Distance from baseline to underline position
UnderlineThickness	Underline thickness
Weight	Font weight (light, bold, etc.)
XHeight	(Western) x-height relative to em size.

VectorCollection Class

Method	Description
Add	
AddRange	
Clear	
CloneCore	
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	
Copy	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetRange	
GetType	Gets the Type of the current instance. Inherited from Object.
IAddChild.AddChild	
IAddChild.AddText	
IList.Add	
IList.Contains	
IList.IndexOf	
IList.Insert	
IList.Remove	
IndexOf	
Insert	
InsertRange	
LastIndexOf	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.

MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
RemoveRange	
SetRange	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
VectorCollection	
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Capacity	
Count	
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a

UIContext

complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

VectorCollectionConverter Class

Definition: VectorCollectionConverter - Converter class for converting instances of other types to and from VectorCollection instances.

Method	Description
CanConvertFrom	CanConvertFrom - Returns whether or not this class can convert from a given type.
CanConvertFrom	Inherited from TypeConverter.
CanConvertTo	Inherited from TypeConverter.
CanConvertTo	CanConvertTo - Returns whether or not this class can convert to a given type.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	ConvertFrom - Attempt to convert to a VectorCollection from the given object
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	ConvertTo - Attempt to convert a VectorCollection to the given type
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.

SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.
VectorCollectionConverter	

VideoData Class

Definition: Enables playing of video files according to the state of a time node.

Method	Description
BeginIn	Schedules an interactive begin time. Inherited from MediaData.
CloneCore	Clones this MediaData.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable. Inherited from MediaData.
Dispose	Dispose the object. Inherited from MediaData.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null. Inherited from MediaData.
EndIn	Schedules an interactive end time. Inherited from MediaData.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Finalizes the MediaData. Inherited from MediaData.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore. Inherited from MediaData.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.

OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
Pause	Pauses this media. Inherited from MediaData.
Play	Begins playback of media. Inherited from MediaData.
PropagateEventHandler	Propogates event handler to the timeline Inherited from MediaData.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this media. Inherited from MediaData.
Seek	Moves the timeline for this media. Inherited from MediaData.
ToString	Persist MediaData in a string Inherited from MediaData.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
VideoData	Initializes a new instance of the VideoData class.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Accesses the Acceleration SMIL attribute. Inherited from MediaData.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Accesses the AutoReverse SMIL attribute. Inherited from MediaData.
Begin	Accesses the Begin SMIL attribute. Inherited from MediaData.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	The current repetition iteration. Inherited from MediaData.
CurrentTime	The current time local to this media. Inherited from MediaData.
Deceleration	Accesses the Deceleration SMIL attribute. Inherited from MediaData.
Duration	Accesses the Duration SMIL attribute. Inherited from MediaData.
End	Accesses the End SMIL attribute. Inherited from MediaData.
EndSync	Accesses the EndSync SMIL attribute. Inherited from MediaData.
Fill	Accesses the Fill SMIL attribute. Inherited from MediaData.
FillDefault	Accesses the FillDefault SMIL attribute. Inherited from MediaData.
HasAudio	True if the media has audio output. Inherited from MediaData.
HasChanged	True if the media has changed since the last tick. Inherited from

	MediaData.
HasVideo	True if the media has a visual output. Inherited from MediaData.
Height	Get the video Height in pixels Inherited from MediaData.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	True if the media is active, false otherwise. Inherited from MediaData.
IsEnabled	True if the media is enabled, false otherwise. Inherited from MediaData.
IsForwardProgressing	True if the media is moving from past to future. Inherited from MediaData.
IsOverridingBaseValue	True if the media is either changing or in a fill state, false otherwise. Inherited from MediaData.
IsPaused	True if this media is paused. Inherited from MediaData.
IsReversed	True if this media is in a reverse period. Inherited from MediaData.
MediaDuration	Returns the native media duration. Inherited from MediaData.
Mute	Accesses the mute state of media playback. Inherited from MediaData.
ParentTimeline	Accesses the ParentTimeline attribute. Inherited from MediaData.
Progress	The current progress of the media, from 0 to 1. Inherited from MediaData.
RepeatCount	Accesses the RepeatCount SMIL attribute. Inherited from MediaData.
RepeatDuration	Accesses the RepeatDuration SMIL attribute. Inherited from MediaData.
Restart	Accesses the Restart SMIL attribute. Inherited from MediaData.
RestartDefault	Accesses the RestartDefault SMIL attribute. Inherited from MediaData.
Speed	Accesses the Speed SMIL attribute. Inherited from MediaData.
State	Returns the current state of the media. Inherited from MediaData.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
Volume	Accesses the volume of media playback. Inherited from MediaData.
Width	Get the video Width in pixels Inherited from MediaData.

VideoDataConverter Class

Definition: VideoDataConverter

Method	Description
CanConvertFrom	CanConvertFrom
CanConvertFrom	Inherited from TypeConverter.

CanConvertTo	Inherited from TypeConverter.
CanConvertTo	TypeConverter method override.
ConvertFrom	Inherited from TypeConverter.
ConvertFrom	ConvertFromString
ConvertFromInvariantString	Inherited from TypeConverter.
ConvertFromString	Inherited from TypeConverter.
ConvertTo	Inherited from TypeConverter.
ConvertTo	TypeConverter method implementation.
ConvertToInvariantString	Inherited from TypeConverter.
ConvertToString	Inherited from TypeConverter.
CreateInstance	Inherited from TypeConverter.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetConvertFromException	Inherited from TypeConverter.
GetConvertToException	Inherited from TypeConverter.
GetCreateInstanceSupported	Inherited from TypeConverter.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetProperties	Inherited from TypeConverter.
GetPropertiesSupported	Inherited from TypeConverter.
GetStandardValues	Inherited from TypeConverter.
GetStandardValuesExclusive	Inherited from TypeConverter.
GetStandardValuesSupported	Inherited from TypeConverter.
GetType	Gets the Type of the current instance. Inherited from Object.
IsValid	Inherited from TypeConverter.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SortProperties	Inherited from TypeConverter.
ToString	Returns a String that represents the current Object. Inherited from Object.
VideoDataConverter	

Visual Class

Definition: Base class for all Visual types. It provides services and properties common to all Visuals, including hit-testing, coordinate transformation, and bounding box calculations.

Method	Description
ClearValue	Clears the local value of a property Inherited from DependencyObject.
Equals	Determines whether two Object instances are equal. Inherited from Object.

Finalize	Releases all resources held by the Visual object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetLocalValueEnumerator	Create a local value enumerator for this instance Inherited from DependencyObject.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Retrieve the value of a property Inherited from DependencyObject.
HitTestCore	HitTestCore implements whether we have hit the bounds of this visual.
InvalidateProperty	Invalidates a property Inherited from DependencyObject.
IVisual.FindCommonVisualAncestor	
IVisual.HitTest	
IVisual.IsAncestorOf	
IVisual.IsDescendantOf	
IVisual.TransformFromAncestor	
IVisual.TransformFromDescendant	
IVisual.TransformFromVisual	
IVisual.TransformToAncestor	
IVisual.TransformToDescendant	
IVisual.TransformToVisual	
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
OnDelayedInvalidate	TODO: Left over from WCP FastBuild, determine relevance in future version of FastBuild Inherited from DependencyObject.
OnPropertyInvalidated	Notification that a specified property has been invalidated Inherited from DependencyObject.
ReadLocalValue	Retrieve the local value of a property (if set) Inherited from DependencyObject.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
SetValue	Sets the local value of a property Inherited from DependencyObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateProperty	Retrieve the value of a property (for use by native cache backed custom get accessors) Inherited from DependencyObject.
ValidatePropertyCore	Allows subclasses to participate in property value computation Inherited from DependencyObject.
Visual	Creates a new instance of the Visual class.

Property	Description
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
DependencyObjectType	Returns the DType that represents the CLR type of this instance Inherited from DependencyObject.

HitTestBounds	HitBounds returns the hit region bounding box for the current visual.
IsDisposed	Gets a value that indicates whether the system has disposed of the Visual.

This example demonstrates how to use a `PrintContext` object to write to a printer. The following code demonstrates how to print a `Visual` (the `VectorPage` in this example) to the default printer. The `VectorPage` in this example derives from `DrawingVisual` and is used to draw several shapes. Two `VectorPage` objects are printed by calling the `AddPage` method twice.

[C#]

```
public static void PrintPaginatedVisualToDefaultPrinter() {
    // create a print context for my default printer
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext();

    // Add two pages to the default rendition
    pc.AddPage(new VectorPage());
    pc.AddPage(new VectorPage());

    //Print
    pc.Print();
}
```

The next example demonstrates how to print a `Visual` to a printer that the user selects. The `SelectPrintQueue` method is used to produce a dialog box that enables the user to select the printer, number of copies to print, and other options.

[C#]

```
public static void PrintPaginatedVisualToSelectedPrinter() {
    // Let user select a printer queue using common dialog box
    System.Printing.PrintSubSystem.PrintQueue queue =
        MSAvalon.Windows.Media.PrintContext.SelectPrintQueue();

    if (queue != null) {
        // Create MSAvalon.Windows.Media.PrintContext if OK button was clicked
        MSAvalon.Windows.Media.PrintContext pc =
            new MSAvalon.Windows.Media.PrintContext(queue, "Printing Example");

        // Add a page of visual
        pc.AddPage(new VectorPage());

        // Print
        pc.Print();
    }
}
```

The next example demonstrates how to print to landscape and portrait page orientations. In the following code, a `JobTicket` is obtained and used to set the page orientation. The `DimensionPage` in this example is a `DrawingVisual` that adjust the size of its drawing to the size of the output page. The `GetPaperWidthHeight` method in this example is used to retrieve the page size information from the `JobTicket`.

[C#]

```
public static void MixedOrientationPrint() {
```

```

System.Printing.PrintSubSystem.LocalPrintServer ps =
    new System.Printing.PrintSubSystem.LocalPrintServer();

System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

MSAvalon.Windows.Media.PrintContext pc =
    new MSAvalon.Windows.Media.PrintContext(queue, "Landscape Example");

Microsoft.Printing.JobTicket.JobTicket jt = pc.JobTicket;

// Print a page with landscape orientation.
{
    jt.PageOrientation.Value =
        System.Printing.Configuration.PrintSchema.OrientationValues.Landscape;

    double width, height;

    GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

    // Add a page of visual
    pc.AddPage(new DimensionPage(height, width));
}

// Print a page with portrait orientation.
{
    jt.PageOrientation.Value =
        System.Printing.Configuration.PrintSchema.OrientationValues.Portrait;

    double width, height;

    GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

    // Add a page of visual
    pc.AddPage(new DimensionPage(width, height));
}

// Print
pc.Print();
}

internal static void GetPaperWidthHeight(Microsoft.Printing.JobTicket.JobTicket jt,
    System.Printing.PrintSubSystem.PrintQueue pq, out double width, out double height) {

    Microsoft.Printing.DeviceCapabilities.DeviceCapabilities devcap =
        new Microsoft.Printing.DeviceCapabilities.DeviceCapabilities(
            pq.AcquireDeviceCapabilities(null));

    devcap.LengthUnitType = System.Printing.Configuration.PrintSchema.LengthUnitTypes.Inch;

    width = (double) devcap.PageCanvasSizeCap.CanvasSizeX * 96;
    height = (double) devcap.PageCanvasSizeCap.CanvasSizeY * 96;
}

```

The next example demonstrates how to print based on the `GetPage` event. The **GeneratePage** method is used to handle the event

[C#]

```
public static void EventDriven() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    // Create MSAvalon.Windows.Media.PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "EventDriven(2 pages)");

    pc.GetPage += new MSAvalon.Windows.Media.PrintContext.GetPageEventHandler(GetAPage);

    // Print
    pc.Print();
}

private static MSAvalon.Windows.Media.Visual GeneratePage(object sender,
    int pageNo, MSAvalon.Windows.Media.GetPageEventArgs ev) {

    if (pageNo == 0) {
        return new VectorPage();
    }
    else if (pageNo == 1) {
        return new ImagePage();
    }
    else return null;
}
```

When running the sample and performing this operation, a dialog box with an assertion error message may appear. Click the Ignore All button to ignore the assertion and print the pages.

The final example shows how to print to a file by using the PrintContext object's Output property.

[C#]

```
public static void PrintToFile() {
    System.Printing.PrintSubSystem.LocalPrintServer ps =
        new System.Printing.PrintSubSystem.LocalPrintServer();

    System.Printing.PrintSubSystem.PrintQueue queue = ps.DefaultPrintQueue;

    // Create PrintContext if OK button was clicked
    MSAvalon.Windows.Media.PrintContext pc =
        new MSAvalon.Windows.Media.PrintContext(queue, "PrintToFile");

    pc.Output = "test.prn";

    // Add a page of visual
    pc.AddPage(new VectorPage());

    // Print
    pc.Print();
}
```

This example demonstrates how to use a `PrintContext` object to print a rendered "Longhorn" markup language (code-named "XAML") file.

In the following example, the `LoadContent` method is used to load, parse, and return a "XAML" file's root `UIElement`. The `UIElement`, `page`, is then configured with a `LayoutManager`, added to the `PrintContext` object using the `AddPage` method, and printed with the `Print` method.

[C#]

```
public static void PrintXamlFile(string xamlFilePath) {
    // Let user select a printer queue using common dialog box
    System.Printing.PrintSubSystem.PrintQueue queue =
        MS Avalon.Windows.Media.PrintContext.SelectPrintQueue();

    if (queue != null) {

        MS Avalon.Windows.Media.PrintContext pc =
            new MS Avalon.Windows.Media.PrintContext(queue, "Element printing test");

        MS Avalon.Windows.UIElement page = LoadContent(xamlFilePath);
        MS Avalon.Windows.LayoutManager lm = new MS Avalon.Windows.LayoutManager(page);

        double width, height;

        GetPaperWidthHeight(pc.JobTicket, queue, out width, out height);

        lm.Size = new MS Avalon.Windows.Size(width, height);
        lm.UpdateLayout();

        pc.AddPage(page);
        pc.Print();

    }
}
```

The `LoadContent` method used in the previous example is used to load the "XAML" file, parse it, and return its root `UIElement`. The `GetPaperWidthHeight` method in this example is used to retrieve the page size information from the `JobTicket`.

[C#]

```
public static MS Avalon.Windows.UIElement LoadContent(string xamlFilePath) {

    System.IO.Stream xamlFileStream = System.IO.File.OpenRead(xamlFilePath);

    MS Avalon.Windows.UIElement root = null;

    try {
        MS Avalon.Windows.Serialization.ParserContext pc =
            new MS Avalon.Windows.Serialization.ParserContext();

        System.Security.PermissionSet ps =
            MS Avalon.Windows.TrustManagement.TrustManager.GetDefaultPermissions();

        System.Security.Permissions.FileIOPermission fiop =
            new System.Security.Permissions.FileIOPermission(
                System.Security.Permissions.FileIOPermissionAccess.AllAccess,
                System.IO.Path.GetFullPath(".\\"));
    }
```

```

        ps.AddPermission(fiop);

        root =
        (MSAvalon.Windows.UIElement) MSAvalon.Windows.Serialization.Parser.LoadXml(
            xamlFileStream, null, pc, null, ps);
    }
    catch (Exception e) {
        Console.WriteLine("Load Failed:");
        Console.WriteLine(e);
    }
    finally {
        // done with the stream
        xamlFileStream.Close();
    }

    return root;
}

internal static void GetPaperWidthHeight(Microsoft.Printing.JobTicket.JobTicket jt,
    System.Printing.PrintSubSystem.PrintQueue pq, out double width, out double height) {

    Microsoft.Printing.DeviceCapabilities.DeviceCapabilities devcap =
        new Microsoft.Printing.DeviceCapabilities.DeviceCapabilities(
            pq.AcquireDeviceCapabilities(null));

    devcap.LengthUnitType = System.Printing.Configuration.PrintSchema.LengthUnitTypes.Inch;

    width = (double) devcap.PageCanvasSizeCap.CanvasSizeX * 96;
    height = (double) devcap.PageCanvasSizeCap.CanvasSizeY * 96;
}

```

VisualCollection Class

Definition: An ordered collection of Visual objects.

Method	Description
Add	Appends a Visual to the end of the VisualCollection.
Clear	Removes all elements from the VisualCollection.
Contains	Returns a Boolean value that indicates whether the passed Visual is contained in the collection.
CopyTo	Copies the current collection into the passed Array.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	Returns an enumerator that can iterate through the VisualCollection.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetTransform	Gets the transform for the specified child.
GetType	Gets the Type of the current instance. Inherited from Object.
IndexOf	Returns the zero-based index of the Visual. If the Visual is not in the VisualCollection -1 is returned.

Insert	Inserts an element into the VisualCollection at the specified index.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	Removes the specified visual from the VisualCollection.
RemoveAt	Removes the Visual at the specified index.
RemoveRange	Removes a range of Visuals from the VisualCollection.
SetTransform	Sets the transform for the specified child.
ToString	Returns a String that represents the current Object. Inherited from Object.
TrimToSize	Sets the capacity to the actual number of elements in the VisualCollection.

Property	Description
Capacity	Gets or sets the number of elements that the VisualCollection can contain.
Count	Gets the number of elements in the collection.
IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread-safe).
Item	Indexer for the VisualCollection. Gets or sets the Visual stored at the zero-based index of the VisualCollection.
SyncRoot	Gets an object that can be used to synchronize access to the ICollection.

VisualManager Class

Definition: Renders a tree of Visual objects to a rendering target, typically a window.

Method	Description
DeviceUnitsFromMeasureUnits	Transforms a point in measure units to a point in device coordinates.
Dispose	Releases the resources associated with the object.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
HandleMessage	The VisualManager needs to see all windows messages so that it can appropriately react to them.
MeasureUnitsFromDeviceUnits	Transforms a point in device coordinates to a point in measure units.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetContext	Associates this UIContextObject with a UIContext. Inherited from UIContextObject.
ToString	Returns a String that represents the current Object. Inherited from Object.
VisualManager	Attaches a VisualManager to the hWnd

Property	Description
BackgroundColor	This is the color that is drawn before everything else. If this color has an alpha component other than 1 it will be ignored.
Context	Returns the UIContext that this UIContextObject is associated with. Inherited from UIContextObject.
IsDisposed	
RootVisual	Gets and sets the root Visual of this VisualManager.

The VisualManager is not thread-safe. Accessing the VisualManagerManager from a thread other than the thread in which it was created will throw an InvalidOperationException.

INTERFACES

IHyphenate Interface

Definition: IHyphenate is the interface for Hyphenation Service Provider

Method	Description
Hyphenate	Return possible HyphenationCandidate for a given word. Use passed in culture. The service provider will check compatibility before it proceed to find hyphenation candidates.

IRetainedRender Interface

Definition: If this interface is implemented on a class that is derived from a RetainedVisual, the RetainedVisual operations in validation mode, i.e. the graphics sub-system will call OnRender in a lazy fashion. (e.g. if the Visual appears for the first time on the screen). Note that OnRender can be called by the system anytime.

Method	Description
Render	The implementation of RenderCore can use the passed in DrawingContext to render this Visual's content.

IVisual Interface

Definition: This interface defines the common methods and services available from a Visual object.

Method	Description
FindCommonVisualAncestor	Returns the common ancestor of two Visual objects.
HitTest	Return top most visual of a hit test.
IsAncestorOf	Returns true if the Visual is an ancestor of the argument Visual. Inherited from .
IsDescendantOf	Returns true if the Visual is a descendant of the argument Visual. Inherited from .
TransformFromAncestor	Returns a transform that can be used to transform coordinates from the specified ancestor to this node. Inherited from .
TransformFromDescendant	Returns a transform that can be used to transform coordinate from the descendant node to this Visual. Inherited from .
TransformFromVisual	The returned matrix can be used to transform coordinates from the specified Visual to the this Visual. Inherited from .
TransformToAncestor	Returns a transformation matrix that can be used to transform coordinates

	from this node to a specified ancestor Visual. Inherited from .
TransformToDescendant	Returns a transformation matrix that can be used to transform coordinates from this node to a specified descendant Visual. Inherited from .
TransformToVisual	The returned matrix can be used to transform coordinates from this Visual to the specified Visual. Inherited from .

Property	Description
Children	Gets a collection of the Visual's children. Inherited from .
Clip	Gets or sets the clipping region of this Visual. Inherited from .
HasChildren	Gets a value that indicates whether the Visual has a child collection. Inherited from .
Opacity	Gets or sets the opacity of the Visual. Inherited from .
Parent	Gets the parent Visual. Inherited from .
VisualContentBounds	VisualContentBounds returns the bounding box for the contents of the current visual.
VisualDescendantBounds	VisualDescendantBounds returns the union of all of the content bounding boxes for all of the descendants of the current visual, but not including the contents of the current visual.

MSAvalon.Wind ws.Media.Animation

The following tables list the members exposed by the MSAvalon.Windows.Media.Animation namespace.

Classes

Animatable	Any class that doesn't derive from DependencyObject but which has properties that can be animated should derive from this class.
AnimationCollection	This abstract class provides base functionality for animation collections, such as ColorAnimationCollection, DoubleAnimationCollection, and SizeAnimationCollection.
AnimationEffect	Override this class to implement element level animations which can participate in the rendering process to instantiate animations on multiple elements at rendering time.
AnimationEffectCollection	Holds a collection of AnimationEffects.
BoolAnimationCollection	Represents a collection of BoolModifier animations.
BoolModifier	
BoolTimedModifier	
ByteAnimationCollection	Represents a collection of BoolModifier animations.
ByteModifier	
ByteTimedModifier	
CharAnimationCollection	Represents a collection of CharModifier animations.
CharModifier	
CharTimedModifier	
ColorAnimation	Animates a color value of a property.
ColorAnimationCollection	Represents a collection of ColorModifier animations.
ColorKeyFrameCollection	
ColorModifier	
ColorTimedModifier	
DecimalAnimationCollection	Represents a collection of DecimalModifier animations.
DecimalModifier	
DecimalTimedModifier	
DoubleAnimation	Used to animate properties that accept a Double value.
DoubleAnimationCollection	Represents a collection of DoubleModifier animations.
DoubleKeyFrameCollection	
DoubleModifier	
DoubleTimedModifier	
FloatAnimation	Used to animate properties that accept a Single value.
FloatAnimationCollection	Represents a collection of FloatModifier animations.
FloatKeyFrameCollection	
FloatModifier	
FloatTimedModifier	
IntAnimationCollection	Represents a collection of IntModifier animations.
IntModifier	
IntTimedModifier	
LengthAnimation	Used to animate properties that accept a Length value.

LengthAnimationCollection	Represents a collection of LengthModifier animations.
LengthKeyFrameCollection	
LengthModifier	
LengthTimedModifier	
LongAnimationCollection	Represents a collection of LongModifier animations.
LongModifier	
LongTimedModifier	
MatrixAnimationCollection	Represents a collection of MatrixModifier animations.
MatrixModifier	
MatrixTimedModifier	
Modifier	
ObjectAnimationCollection	Represents a collection of ObjectModifier animations.
ObjectModifier	
ObjectTimedModifier	
PathAnimation	This animation can be used inside of a MatrixAnimationCollection to move a visual object along a path.
PointAnimation	Used to animate properties that accept Point values.
PointAnimationCollection	Represents a collection of PointModifier animations.
PointKeyFrameCollection	
PointModifier	
PointTimedModifier	
RectAnimation	Used to animate properties that accept a Rect value.
RectAnimationCollection	Represents a collection of RectModifier animations.
RectKeyFrameCollection	
RectModifier	
RectTimedModifier	
ShortAnimationCollection	Represents a collection of ShortModifier animations.
ShortModifier	
ShortTimedModifier	
SizeAnimation	Defines an animation based on the Size of an object. By providing Size information, an object can appear to shrink or enlarge over a period of time.
SizeAnimationCollection	Represents a collection of SizeModifier animations.
SizeKeyFrameCollection	
SizeModifier	
SizeTimedModifier	
StringAnimationCollection	Represents a collection of StringModifier animations.
StringModifier	
StringTimedModifier	
Timeline	Maintains run-time timing state for timed objects.
TimelineBuilder	An object that can be used to create Timeline objects.
TimeManager	The object that controls an entire timing tree.
TimeSyncValueTypeConverter	An object that performs type conversions involving TimeSyncValue values.
TimeTypeConverter	An object that performs type conversions involving Time values.
VectorAnimation	Used to animate properties that accept a Vector value.
VectorAnimationCollection	Represents a collection of VectorModifier animations.

VectorKeyFrameCollection

VectorModifier

VectorTimedModifier

Interfaces

IClock

Represents an object that can provide linear time values.

IModifier

Defines the basic behavior of a modifier object. A modifier is an object that takes an object, called the *base value*, of a certain type and returns another object of the same type as its output.

ITimingControl

Defines the behavior of timelines and timed objects.

ITimingControlBuilder

Represents an object that can build a timeline template.

Enumerations

AnimationType

Describes the behavior of an animation.

CloneType

The types of clones that CloneCore may request.

InterpolationMethod

Describes how an animation calculates its output values.

KeyTimeType

The different types of KeyTimes

TimeEndSync

Values for the endSync attribute, which specifies how a container calculates its simple duration based on the children's durations.

TimeFill

Specifies how the timeline behaves after it is no longer active.

TimeRestart

Values for the Timeline.Restart attribute.

TimeSeekOrigin

Indicates a timeline position; used to specify the behavior of the ITimingControl interface's Seek method by defining the position to which the offset is applied.

TimeSyncBase

The event to synchronize a begin or end value to.

Structures

ColorKeyFrame

DoubleKeyFrame

FloatKeyFrame

KeySpline

This class is used to pass an array of key splines into the KeySplines property of an animation fragment.

KeyTime

A KeyTime is use to specify when relative to the time of an animation that a KeyFrame takes place.

LengthKeyFrame

PointKeyFrame

RectKeyFrame

SizeKeyFrame

Time

A value representing time, with associated time arithmetic operations.

TimelineEnumerator

Enumerates items in an TimelineList collection.

TimeSyncValue

A value representing an absolute or relative begin or end time for a timeline.

VectorKeyFrame

MSAvalon.Windows.Media.Animation

CLASSES

Animatable Class

Definition: Any class that doesn't derive from DependencyObject but which has properties that can be animated should derive from this class.

Method	Description
Animatable	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneCore	Subclasses must implement this to provide clones of themselves.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DisableCore	
EmbeddedAnimationCollectionReader	Maybe this should be internal??
EmbeddedAnimationCollectionWriter	
EmbeddedChangeableReader	
EmbeddedChangeableWriter	
EnableCore	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentValue	Returns a non-animated version of this Animatable that represents its current state.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive

	from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	This will change the time parent for animations on any of the properties of this Changeable which are inheriting their time parent.
SetDefaultParentTimelineCore	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
DefaultParentTimeline	The current parent Timeline associated with this Animatable. This will be the Timeline set to the ParentTimeline property of this Animatable if one has been set and if not, the Timeline last passed into the SetDefaultParentTimeline method.
HasAnimations	Returns true if any of the properties on this Changeable have animations attached.
IsAnimating	Returns true if any of the properties on this Changeable have animations attached that are currently animating their value.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsOverridingBaseValue	Returns true if any of the properties on this Changeable have animations attached that are currently affecting their value either by animating it or holding it in a fill state.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a

UIContext

DrawingContext command. Inherited from Changeable.

Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

AnimationCollection Class

Definition: This abstract class provides base functionality for animation collections, such as ColorAnimationCollection, DoubleAnimationCollection, and SizeAnimationCollection.

Method	Description
AnimationCollection	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Disable	
DisableImpl	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	
EnableImpl	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator
GetEnumeratorImpl	
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Calculates and returns the output value of the animation collection.
GetValueImpl	Subclasses implement this to provide a Modifier at a given index.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified

	Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
<code>OnChanged</code>	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
<code>PropagateEventHandler</code>	Shares a <code>Changed</code> event handler with the current object's data members or removes it. Inherited from <code>Changeable</code> .
<code>ReadPreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
<code>ReferenceEquals</code>	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
<code>SetDefaultParentTimeline</code>	
<code>SetValueImpl</code>	Subclasses implement this to set a <code>Modifier</code> at a given index.
<code>ToString</code>	Returns a <code>String</code> that represents the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ValidateObjectState</code>	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
<code>WritePostscript</code>	Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code> .
<code>WritePreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from <code>Changeable</code> .

Property	Description
<code>AllowChangeableReferenceOverride</code>	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
<code>AnimationType</code>	<code>AnimationType</code>
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>Count</code>	
<code>CountImpl</code>	
<code>IsChangeable</code>	Gets a <code>Boolean</code> that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
<code>IsChanging</code>	Returns true if at least one of the animations in the animation list is currently active.
<code>IsOverridingBaseValue</code>	Returns true if at least one of the animations in the animation list is currently on.
<code>IsUsingBaseValue</code>	
<code>IsUsingBaseValueImpl</code>	
<code>Item</code>	Use this to get or set a <code>Modifier</code> at a given index.
<code>StatusOfNextUse</code>	Gets or sets a <code>UseStatus</code> enumeration that specifies how the <code>Changeable</code> object behaves when it is "used." A <code>Changeable</code> object is considered used in the following situations: the object is set into a <code>Property System</code> property, the object is used as a sub-object in a

	complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The AnimationCollection classes provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate a base value. When a property requests the current value of an AnimationCollection, the property calls the AnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate a property using "Longhorn" markup language (code-named "XAML"). To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the RectangleWidth of a Rectangle, the RectangleWidth must be set to a non-animated value (in this case, a Length object).

In the following example, the Width of a Button is animated. Because the Width property takes a Length, a LengthAnimation is needed. A LengthAnimationCollection is used to contain the LengthAnimation. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no WidthAnimations property, the LengthAnimationCollection is associated directly with the button's Width property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

<Button.Width>
  <LengthAnimationCollection>
    <LengthAnimation To="50" Duration="5" RepeatCount="500"
      AutoReverse="True"/>
  </LengthAnimationCollection>
</Button.Width>

  A Button
</Button>
```

In the next example, the Background color of a second button is animated. The Background property takes a Brush. In this example, a SolidColorBrush is used to fill the button's Background, although a gradient, image, or pattern could have been used. To animate the button's background color, the Color of the SolidColorBrush must be animated. Because the SolidColorBrush.Color property accepts a Color, a ColorAnimation is used to animate the property. The SolidColorBrush.Color property has a corresponding ColorAnimations property, so the ColorAnimation is nested within the ColorAnimations property in order to animate the color of the brush.

```
<Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">
```



```

<Button.Background>
  <SolidColorBrush Color="Blue">
    <SolidColorBrush.ColorAnimations>
      <ColorAnimation From="Red" To="Blue" Duration="7"
        RepeatCount="500" AutoReverse="True"/>
    </SolidColorBrush.ColorAnimations>
  </SolidColorBrush>
</Button.Background>

```

```

Another Button
</Button>

```

```

</Canvas>

```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag. For more information about animating properties, see *Animation in "Avalon"*.

AnimationEffect Class

Definition: Override this class to implement element level animations which can participate in the rendering process to instantiate animations on multiple elements at rendering time.

Method	Description
<code>AnimationEffect</code>	Default constructor of an <code>AnimationEffect</code> .
<code>AttachImpl</code>	Override this method if you would like to store per <code>Element</code> instance data for this <code>AnimationEffect</code> . Keep in mind that when this is called the element might not be in an element tree or its children may not have been added. If these things are important to you, you should override this method and sign up for the event notifications that are important to you and update the state of the animation effect when those events are raised.
<code>CloneCore</code>	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .
<code>CloneDownToUnchangeable</code>	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .
<code>Copy</code>	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
<code>DetachImpl</code>	Override this method if you need to do some cleanup when this <code>AnimationEffect</code> is detached from an <code>Element</code> .
<code>EmbeddedChangeableReader</code>	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
<code>EmbeddedChangeableWriter</code>	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
<code>Equals</code>	Determines whether two <code>Object</code> instances are equal. Inherited from

	Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
Invalidate	Calling this method will place this AnimationEffect in the list of animation effects to be processed during the next render. After calling this, the IsInvalid property will return true until the beginning of the next render. If no render is currently scheduled calling this method will schedule a render.
InvalidatePassive	Calling this method will place this AnimationEffect in the list of animation effects to be processed during the next render. After calling this, the IsInvalid property will return true until the beginning of the next render. If no render is scheduled, calling this method will not schedule a new render. Something else will have to trigger a render.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PostLayoutReadImpl	This method will be called once per render pass. This method can be used to read the final values from the element tree after rendering. The element tree will be locked when this method is called.
PreLayoutReadImpl	This method will be called one or more times when the rendering process begins. The element tree will be locked when this method is called.
PreLayoutWriteImpl	This method will be called one or more times when the rendering process begins and after OnPreLayoutRead. The element tree will be unlocked so so this is the time to add animations or new elements. Tip: It's best to do as little reading from the element tree as possible in this method. Do all the reading you can in the OnPreLayoutRead method.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an

	invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsInvalid	Returns true if this AnimationEffect is in the list of AnimationEffects to be processed during the next render.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Target	The element that this AnimationEffect is attached to or null if the AnimationEffect is not attached to an element.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

AnimationEffectCollection Class

Definition: Holds a collection of AnimationEffects.

Method	Description
Add	
AnimationEffectCollection	Creates an empty AnimationEffectCollection with a default capacity for a single animation.
Clear	
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	
Copy	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and

	returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Pr erty

Description

AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Empty	An unchangeable empty AnimationEffectCollection.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsFixedSize	
IsReadOnly	
IsSynchronized	
Item	this - typed version of indexer
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

BoolAnimationCollection Class

Definition: Represents a collection of BoolModifier animations.

Method	Description
Add	The Add(BoolModifier) and Add(Object) methods add animations to the collection. The Add(Boolean,BoolAnimationCollection) method calculates the current value of the specified collection based on the specified base value.
AddChild	Implementation of AddChild. Adds a Modifier to this AnimationCollection from Markup.
AddText	Implementation of AddText. This is not implemented on this class.
Apply	Implementation of Apply. Applies an animation collection in markup to an element.
BoolAnimationCollection	Creates an empty BoolAnimationCollection with a default capacity for a single animation.
Clear	Clears the collection by setting the collection's Count to 0.
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a Boolean that indicates whether the collection contains the specified BoolModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this BoolAnimationCollection.

CopyTo	Copies the entire BoolAnimationCollection to the specified one-dimensional array, starting at the specified index of the target array.
Disable	Inherited from AnimationCollection.
DisableImpl	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Inherited from AnimationCollection.
EnableImpl	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator Inherited from AnimationCollection.
GetEnumeratorImpl	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Uses the specified base value to calculate and return the animation collection's current value.
GetValueImpl	Provides a Modifier at a given index.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Addition	
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	

RemoveAt	
SetDefaultParentTimeline	Inherited from AnimationCollection.
SetValueImpl	Sets a Modifier at a given index.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AnimationType	AnimationType
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Inherited from AnimationCollection.
CountImpl	
Empty	An unchangeable empty BoolAnimationCollection.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Returns true if at least one of the animations in the animation list is currently active. Inherited from AnimationCollection.
IsFixedSize	
IsOverridingBaseValue	Returns true if at least one of the animations in the animation list is currently on. Inherited from AnimationCollection.
IsReadOnly	
IsSynchronized	
IsUsingBaseValue	Inherited from AnimationCollection.
IsUsingBaseValueImpl	
Item	this - typed version of indexer
Item	Use this to get or set a Modifier at a given index. Inherited from AnimationCollection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The AnimationCollection classes, such as BoolAnimationCollection, provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate a base value. When a property requests the current value of an BoolAnimationCollection, the property calls the BoolAnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate a property using "Longhorn" markup language (code-named "XAML"). To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the RectangleWidth of a Rectangle, the RectangleWidth must be set to a non-animated value (in this case, a Length object).

In the following example, the Width of a Button is animated. Because the Width property takes a Length, a LengthAnimation is needed. A LengthAnimationCollection is used to contain the LengthAnimation. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no WidthAnimations property, the LengthAnimationCollection is associated directly with the button's Width property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

<Button.Width>
  <LengthAnimationCollection>
    <LengthAnimation To="50" Duration="5" RepeatCount="500"
      AutoReverse="True"/>
  </LengthAnimationCollection>
</Button.Width>

A Button
</Button>
```

In the next example, the Background color of a second button is animated. The Background property takes a Brush. In this example, a SolidColorBrush is used to fill the button's Background, although a gradient, image, or pattern could have been used. To animate the button's background color, the Color of the SolidColorBrush must be animated. Because the SolidColorBrush.Color property accepts a Color, a ColorAnimation is used to animate the property. The SolidColorBrush.Color property has a corresponding ColorAnimations property, so the ColorAnimation is nested within the ColorAnimations property in order to animate the color of the brush.

```
<Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">

<Button.Background>
  <SolidColorBrush Color="Blue">
    <SolidColorBrush.ColorAnimations>
      <ColorAnimation From="Red" To="Blue" Duration="7"
        RepeatCount="500" AutoReverse="True"/>
    </SolidColorBrush.ColorAnimations>
  </SolidColorBrush>
</Button.Background>
```



```

        </SolidColorBrush.ColorAnimations>
    </SolidColorBrush>
</Button.Background>

```

```

    Another Button
</Button>

```

```

</Canvas>

```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag. For more information about animating properties, see *Animation in "Avalon"*.

BoolModifier Class

Method	Description
<code>BoolModifier</code>	
<code>CloneCore</code>	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .
<code>CloneDownToUnchangeable</code>	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .
<code>Copy</code>	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
<code>Copy</code>	Creates a copy of this <code>BoolModifier</code>
<code>EmbeddedChangeableReader</code>	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
<code>EmbeddedChangeableWriter</code>	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
<code>Equals</code>	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
<code>Finalize</code>	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
<code>GetHashCode</code>	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
<code>GetType</code>	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
<code>GetValue</code>	
<code>GetValueImpl</code>	
<code>IModifier.GetValue</code>	Inherited from <code>Modifier</code> .
<code>MakeUnchangeable</code>	Makes an object immutable; after this method is called on a

	Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a BoolModifier
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from Modifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from Modifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.

UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

BoolTimedModifier Class

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current time.
BoolTimedModifier	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this BoolTimedModifier
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this BoolModifier Inherited from BoolModifier.
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null.
EndIn	Schedules an interactive end time.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Inherited from BoolModifier.
GetValueImpl	Inherited from BoolModifier.
IModifier.GetValue	Inherited from Modifier.

MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a BoolModifier Inherited from BoolModifier.
Pause	Pauses this timeline.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this timeline.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration.
Begin	Gets or sets an offset to the start time of the animation.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	Gets the number of the current iteration of the animation.

CurrentTime	Gets the current time value of the animation.
Deceleration	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase.
Duration	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation.
End	Gets or sets the maximum end time of the animation.
EndSync	Not supported. Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set.
Fill	Gets or sets a value that specifies the state of an object when its animation ends.
FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active.
IsEnabled	
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period.
IsPaused	Gets a value that indicates whether the animation is active and paused.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline.
ParentTimeline	Gets or sets the default parent timeline of the animation.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed.
RepeatCount	Gets or sets the number of times an animation should repeat.
RepeatDuration	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, the animation will repeat itself for the length of time specified by this property.
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached.
RestartDefault	Gets or sets the default value of the Restart property of the current animation and its child timelines.
Speed	Gets or sets the relative speed at which time should pass for the animation, compared to its parent timeline.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline
UIContext	Gets the UIContext of the current object. The UIContext is used for

UsesBaseValue

maintaining thread safety. Inherited from Changeable.
UsesBaseValue Inherited from Modifier.

ByteAnimationCollection Class

Definition: Represents a collection of BoolModifier animations.

Method	Description
Add	Adds animations to the collection.
AddChild	Implementation of AddChild. Adds a Modifier to this AnimationCollection from Markup.
AddText	Implementation of AddText. This is not implemented on this class.
Apply	Implementation of Apply. Applies an animation collection in markup to an element.
ByteAnimationCollection	Creates an empty ByteAnimationCollection with a default capacity for a single animation.
Clear	
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this ByteAnimationCollection.
CopyTo	Copies the entire ByteAnimationCollection to the specified one-dimensional array, starting at the specified index of the target array.
Disable	Inherited from AnimationCollection.
DisableImpl	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Inherited from AnimationCollection.
EnableImpl	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator Inherited from AnimationCollection.
GetEnumeratorImpl	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.

GetValue	Returns the current value of the animation.
GetValueImpl	Provides a Modifier at a given index.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Addition	
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
SetDefaultParentTimeline	Inherited from AnimationCollection.
SetValueImpl	Sets a Modifier at a given index.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AnimationType	AnimationType
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Inherited from AnimationCollection.
CountImpl	
Empty	An unchangeable empty ByteAnimationCollection.

IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Returns true if at least one of the animations in the animation list is currently active. Inherited from AnimationCollection.
IsFixedSize	
IsOverridingBaseValue	Returns true if at least one of the animations in the animation list is currently on. Inherited from AnimationCollection.
IsReadOnly	
IsSynchronized	
IsUsingBaseValue	Inherited from AnimationCollection.
IsUsingBaseValueImpl	
Item	this - typed version of indexer
Item	Use this to get or set a Modifier at a given index. Inherited from AnimationCollection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

AnimationCollection provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate a base value. When a property requests the current value of an AnimationCollection, the property calls the AnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate a property using "Longhorn" markup language (code-named "XAML"). To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the RectangleWidth of a Rectangle, the RectangleWidth must be set to a non-animated value (in this case, a Length object).

In the following example, the Width of a Button is animated. Because the Width property takes a Length, a LengthAnimation is needed. A LengthAnimationCollection is used to contain the LengthAnimation. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no WidthAnimations property, the LengthAnimationCollection is associated directly with the button's Width property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">
```



```

<Button.Width>
  <LengthAnimationCollection>
    <LengthAnimation To="50" Duration="5" RepeatCount="500"
      AutoReverse="True"/>
  </LengthAnimationCollection>
</Button.Width>

```

```

A Button
</Button>

```

In the next example, the Background color of a second button is animated. The Background property takes a Brush. In this example, a SolidColorBrush is used to fill the button's Background, although a gradient, image, or pattern could have been used. To animate the button's background color, the Color of the SolidColorBrush must be animated. Because the SolidColorBrush.Color property accepts a Color, a ColorAnimation is used to animate the property. The SolidColorBrush.Color property has a corresponding ColorAnimations property, so the ColorAnimation is nested within the ColorAnimations property in order to animate the color of the brush.

```

<Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">

  <Button.Background>
    <SolidColorBrush Color="Blue">
      <SolidColorBrush.ColorAnimations>
        <ColorAnimation From="Red" To="Blue" Duration="7"
          RepeatCount="500" AutoReverse="True"/>
      </SolidColorBrush.ColorAnimations>
    </SolidColorBrush>
  </Button.Background>

```

```

Another Button
</Button>

```

```

</Canvas>

```

In the previous example, the ColorAnimationCollection tag, <ColorAnimationCollection>, is omitted when animating the brush's color. When animating a designated animation property—properties of the form PropertyNameAnimations, such as the ColorAnimations property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from UIElement—you must nest the animations within an animation collection tag. For more information about animating properties, see Animation in "Avalon".

ByteModifier Class

Method	Description
ByteModifier	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.

Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
Copy	Creates a copy of this <code>ByteModifier</code>
<code>EmbeddedChangeableReader</code>	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
<code>EmbeddedChangeableWriter</code>	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
<code>Equals</code>	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
<code>Finalize</code>	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
<code>GetHashCode</code>	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
<code>GetType</code>	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
<code>GetValue</code>	
<code>GetValueImpl</code>	
<code>IModifier.GetValue</code>	Inherited from <code>Modifier</code> .
<code>MakeUnchangeable</code>	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
<code>MakeUnchangeableCore</code>	<code>MakeUnchangeableCore</code> Inherited from <code>Modifier</code> .
<code>MemberwiseClone</code>	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ModifyHandlerIfChangeable</code>	Adds or removes a <code>Changed</code> event handler to or from the specified <code>Changeable</code> object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
<code>OnChanged</code>	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
<code>op_Implicit</code>	Implicitly creates an <code>AnimationCollection</code> from a <code>ByteModifier</code>
<code>PropagateEventHandler</code>	Shares a <code>Changed</code> event handler with the current object's data members or removes it. Inherited from <code>Changeable</code> .
<code>ReadPreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
<code>ReferenceEquals</code>	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
<code>SetDefaultParentTimeline</code>	<code>SetDefaultParentTimeline</code> Inherited from <code>Modifier</code> .
<code>ToString</code>	Returns a <code>String</code> that represents the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ValidateObjectState</code>	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
<code>WritePostscript</code>	Causes the current object to validate itself and then invokes the

WritePreamble	<p>OnChanged method. Inherited from Changeable.</p> <p>Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.</p>
---------------	---

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from Modifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from Modifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

ByteTimedModifier Class

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current time.
ByteTimedModifier	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this ByteTimedModifier
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this ByteModifier Inherited from ByteModifier.
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from

	Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null.
EndIn	Schedules an interactive end time.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Inherited from ByteModifier.
GetValueImpl	Inherited from ByteModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a ByteModifier Inherited from ByteModifier.
Pause	Pauses this timeline.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this timeline.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an

	invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration.
Begin	Gets or sets an offset to the start time of the animation.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	Gets the number of the current iteration of the animation.
CurrentTime	Gets the current time value of the animation.
Deceleration	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase.
Duration	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation.
End	Gets or sets the maximum end time of the animation.
EndSync	Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set.
Fill	Gets or sets a value that specifies the state of an object when its animation ends.
FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active.
IsEnabled	
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period.
IsPaused	Gets a value that indicates whether the animation is active and paused.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline.
ParentTimeline	Gets or sets the default parent timeline of the animation.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple

	duration that has elapsed.
RepeatCount	Gets or sets the number of times an animation should repeat.
RepeatDuration	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, it will repeat itself for the length of time specified by this property.
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached.
RestartDefault	Gets or sets the default value of the Restart property of the current animation and its child timelines.
Speed	Gets or sets the relative speed at which time should pass for the animation, compared to its parent timeline.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

CharAnimationCollection Class

Definition: Represents a collection of CharModifier animations.

Method	Description
Add	The Add(CharModifier) and Add(Object) methods add animations to the collection. The Add(Char,CharAnimationCollection) method calculates the current value of the specified collection based on the specified base value.
AddChild	Implementation of AddChild. Adds a Modifier to this AnimationCollection from Markup.
AddText	Implementation of AddText. This is not implemented on this class.
Apply	Implementation of Apply. Applies an animation collection in markup to an element.
CharAnimationCollection	Creates an empty CharAnimationCollection with a default capacity for a single animation.
Clear	Clears the collection by setting the collection's Count to 0.
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a Boolean that indicates whether the collection contains the specified CharModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this CharAnimationCollection.
CopyTo	

Disable	Inherited from AnimationCollection.
DisableImpl	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Inherited from AnimationCollection.
EnableImpl	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator Inherited from AnimationCollection.
GetEnumeratorImpl	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Returns the current value of the animation.
GetValueImpl	Provides a Modifier at a given index.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Addition	
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
SetDefaultParentTimeline	Inherited from AnimationCollection.
SetValueImpl	Sets a Modifier at a given index.

ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AnimationType	AnimationType
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Inherited from AnimationCollection.
CountImpl	
Empty	An unchangeable empty CharAnimationCollection.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Returns true if at least one of the animations in the animation list is currently active. Inherited from AnimationCollection.
IsFixedSize	
IsOverridingBaseValue	Returns true if at least one of the animations in the animation list is currently on. Inherited from AnimationCollection.
IsReadOnly	
IsSynchronized	
IsUsingBaseValue	Inherited from AnimationCollection.
IsUsingBaseValueImpl	
Item	this - typed version of indexer
Item	Use this to get or set a Modifier at a given index. Inherited from AnimationCollection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The AnimationCollection classes, such as CharAnimationCollection, provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate

a base value. When a property requests the current value of an CharAnimationCollection, the property calls the CharAnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate a property using "Longhorn" markup language (code-named "XAML"). To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the RectangleWidth of a Rectangle, the RectangleWidth must be set to a non-animated value (in this case, a Length object).

In the following example, the Width of a Button is animated. Because the Width property takes a Length, a LengthAnimation is needed. A LengthAnimationCollection is used to contain the LengthAnimation. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no WidthAnimations property, the LengthAnimationCollection is associated directly with the button's Width property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

<Button.Width>
  <LengthAnimationCollection>
    <LengthAnimation To="50" Duration="5" RepeatCount="500"
      AutoReverse="True"/>
  </LengthAnimationCollection>
</Button.Width>

A Button
</Button>
```

In the next example, the Background color of a second button is animated. The Background property takes a Brush. In this example, a SolidColorBrush is used to fill the button's Background, although a gradient, image, or pattern could have been used. To animate the button's background color, the Color of the SolidColorBrush must be animated. Because the SolidColorBrush.Color property accepts a Color, a ColorAnimation is used to animate the property. The SolidColorBrush.Color property has a corresponding ColorAnimations property, so the ColorAnimation is nested within the ColorAnimations property in order to animate the color of the brush.

```
<Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">

<Button.Background>
  <SolidColorBrush Color="Blue">
    <SolidColorBrush.ColorAnimations>
      <ColorAnimation From="Red" To="Blue" Duration="7"
        RepeatCount="500" AutoReverse="True"/>
    </SolidColorBrush.ColorAnimations>
  </SolidColorBrush>
</Button.Background>
```

```

    Another Button
  </Button>

```

```

</Canvas>

```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

CharModifier Class

Method	Description
<code>CharModifier</code>	
<code>CloneCore</code>	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .
<code>CloneDownToUnchangeable</code>	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .
<code>Copy</code>	Creates a copy of this <code>CharModifier</code>
<code>Copy</code>	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
<code>EmbeddedChangeableReader</code>	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
<code>EmbeddedChangeableWriter</code>	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
<code>Equals</code>	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
<code>Finalize</code>	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
<code>GetHashCode</code>	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
<code>GetType</code>	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
<code>GetValue</code>	
<code>GetValueImpl</code>	
<code>IModifier.GetValue</code>	Inherited from <code>Modifier</code> .
<code>MakeUnchangeable</code>	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from <code>Changeable</code> .
<code>MakeUnchangeableCore</code>	<code>MakeUnchangeableCore</code> Inherited from <code>Modifier</code> .
<code>MemberwiseClone</code>	Creates a shallow copy of the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ModifyHandlerIfChangeable</code>	Adds or removes a <code>Changed</code> event handler to or from the specified

	Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
<code>OnChanged</code>	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
<code>op_Implicit</code>	Implicitly creates an <code>AnimationCollection</code> from a <code>CharModifier</code>
<code>PropagateEventHandler</code>	Shares a <code>Changed</code> event handler with the current object's data members or removes it. Inherited from <code>Changeable</code> .
<code>ReadPreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
<code>ReferenceEquals</code>	Determines whether the specified <code>Object</code> instances are the same instance. Inherited from <code>Object</code> .
<code>SetDefaultParentTimeline</code>	<code>SetDefaultParentTimeline</code> Inherited from <code>Modifier</code> .
<code>ToString</code>	Returns a <code>String</code> that represents the current <code>Object</code> . Inherited from <code>Object</code> .
<code>ValidateObjectState</code>	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
<code>WritePostscript</code>	Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code> .
<code>WritePreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from <code>Changeable</code> .

Property	Description
<code>AllowChangeableReferenceOverride</code>	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>IsChangeable</code>	Gets a <code>Boolean</code> that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
<code>IsChanging</code>	Gets a value that indicates whether the animation is active. Inherited from <code>Modifier</code> .
<code>IsOverridingBaseValue</code>	Gets a value that indicates whether the animation is active or in a fill period. Inherited from <code>Modifier</code> .
<code>StatusOfNextUse</code>	Gets or sets a <code>UseStatus</code> enumeration that specifies how the <code>Changeable</code> object behaves when it is "used." A <code>Changeable</code> object is considered used in the following situations: the object is set into a <code>Property System</code> property, the object is used as a sub-object in a complex <code>Changeable</code> object, or the object is used in a <code>DrawingContext</code> command. Inherited from <code>Changeable</code> .
<code>UIContext</code>	Gets the <code>UIContext</code> of the current object. The <code>UIContext</code> is used for maintaining thread safety. Inherited from <code>Changeable</code> .
<code>UsesBaseValue</code>	<code>UsesBaseValue</code> Inherited from <code>Modifier</code> .

CharTimedModifier Class

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current time.
CharTimedModifier	
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this CharModifier Inherited from CharModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a copy of this CharTimedModifier
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null.
EndIn	Schedules an interactive end time.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Inherited from CharModifier.
GetValueImpl	Inherited from CharModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified

	object is not modifiable—if its <code>IsChangeable</code> property is false—this method has no effect. Inherited from <code>Changeable</code> .
<code>OnChanged</code>	Called when the current object is modified. Classes that derive from <code>Changed</code> should call this method after they have been modified. Inherited from <code>Changeable</code> .
<code>op_Implicit</code>	Implicitly creates an <code>AnimationCollection</code> from a <code>CharModifier</code> Inherited from <code>CharModifier</code> .
<code>Pause</code>	Pauses this timeline.
<code>PropagateEventHandler</code>	
<code>ReadPreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from <code>Changeable</code> .
<code>ReferenceEquals</code>	Determines whether the specified Object instances are the same instance. Inherited from <code>Object</code> .
<code>Resume</code>	Resumes this timeline.
<code>Seek</code>	Moves the current position of the animation backwards or forwards from either the current time, the <code>Begin</code> time, or the <code>End</code> time.
<code>SetDefaultParentTimeline</code>	<code>SetDefaultParentTimeline</code> Inherited from <code>Modifier</code> .
<code>ToString</code>	Returns a <code>String</code> that represents the current Object. Inherited from <code>Object</code> .
<code>ValidateObjectState</code>	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from <code>Changeable</code> .
<code>WritePostscript</code>	Causes the current object to validate itself and then invokes the <code>OnChanged</code> method. Inherited from <code>Changeable</code> .
<code>WritePreamble</code>	Ensures that simple (non- <code>Changeable</code>) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from <code>Changeable</code> .

Property	Description
<code>Acceleration</code>	Gets or sets the fraction of the simple duration spent in the acceleration phase.
<code>AllowChangeableReferenceOverride</code>	Used in conjunction with the <code>ChangeableUsageOverride</code> type sent in as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
<code>AutoReverse</code>	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration.
<code>Begin</code>	Gets or sets an offset to the start time of the animation.
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>CurrentRepeat</code>	Gets the number of the current iteration of the animation.
<code>CurrentTime</code>	Gets the current time value of the animation.
<code>Deceleration</code>	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase.
<code>Duration</code>	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation.
<code>End</code>	Gets or sets the maximum end time of the animation.

EndSync	Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set.
Fill	Gets or sets a value that specifies the state of an object when its animation ends.
FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active.
IsEnabled	
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period.
IsPaused	Gets a value that indicates whether the animation is active and paused.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline.
ParentTimeline	Gets or sets the default parent timeline of the animation.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed.
RepeatCount	Gets or sets the number of times an animation should repeat.
RepeatDuration	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, it will repeat itself until the time specified by this property.
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached.
RestartDefault	Gets or sets the default value of the Restart property of the current animation and its child timelines.
Speed	Gets or sets the relative speed at which time should pass for the animation, compared to its parent timeline.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

ColorAnimation Class

Definition: Animates a color value of a property.

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current

	time. Inherited from ColorTimedModifier.
CloneCore	Implementation of CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorAnimation	Initializes a new instance of the ColorAnimation class.
Copy	Creates a copy of this ColorModifier Inherited from ColorModifier.
Copy	Creates a copy of this ColorTimedModifier Inherited from ColorTimedModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable. Inherited from ColorTimedModifier.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null. Inherited from ColorTimedModifier.
EndIn	Schedules an interactive end time. Inherited from ColorTimedModifier.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	Inherited from ColorTimedModifier.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Calculates the current value of the animation.
GetValueImpl	Inherited from ColorModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from

	Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a ColorModifier Inherited from ColorModifier.
Pause	Pauses this timeline. Inherited from ColorTimedModifier.
PropagateEventHandler	PropagateEventHandler
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this timeline. Inherited from ColorTimedModifier.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time. Inherited from ColorTimedModifier.
SetDefaultParentTimeline	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase. Inherited from ColorTimedModifier.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration. Inherited from ColorTimedModifier.
Begin	Gets or sets an offset to the start time of the animation. Inherited from ColorTimedModifier.
By	Gets or sets the total amount by which the animation changes its starting value.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	Gets the number of the current iteration of the animation. Inherited from ColorTimedModifier.
CurrentTime	Gets the current time value of the animation. Inherited from ColorTimedModifier.
Deceleration	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase. Inherited from ColorTimedModifier.

Duration	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation. Inherited from ColorTimedModifier.
End	Gets or sets the maximum end time of the animation. Inherited from ColorTimedModifier.
EndSync	Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set. Inherited from ColorTimedModifier.
Fill	Gets or sets a value that specifies the state of an object when its animation ends. Inherited from ColorTimedModifier.
FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines. Inherited from ColorTimedModifier.
From	Gets or sets the starting value of an animation.
InterpolationMethod	Gets or sets a value that specifies how output values are calculated for the animation.
IsAdditive	IsAdditive
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from ColorTimedModifier.
IsCumulative	IsCumulative
IsEnabled	Inherited from ColorTimedModifier.
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future. Inherited from ColorTimedModifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from ColorTimedModifier.
IsPaused	Gets a value that indicates whether the animation is active and paused. Inherited from ColorTimedModifier.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline. Inherited from ColorTimedModifier.
KeyFrames	KeyValues
ParentTimeline	Gets or sets the default parent timeline of the animation. Inherited from ColorTimedModifier.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed. Inherited from ColorTimedModifier.
RepeatCount	Gets or sets the number of times an animation should repeat. Inherited from ColorTimedModifier.
RepeatDuration	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, it will repeat itself for the length of time specified by this property. Inherited from ColorTimedModifier.
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached. Inherited from ColorTimedModifier.
RestartDefault	Gets or sets the default value of the Restart property of the current animation and its child timelines. Inherited from ColorTimedModifier.
Speed	Gets or sets the relative speed at which time should pass for the

	animation, compared to its parent timeline. Inherited from ColorTimedModifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline Inherited from ColorTimedModifier.
To	Gets or sets the ending value of the animation.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	

This example shows how to animate the Fill color of a Ellipse.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml"
  xmlns:def="Definition">

  <Ellipse CenterX="200" CenterY="200" RadiusX = "100" RadiusY = "100" >
  <Ellipse.Fill>
    <SolidColorBrush>
      <SolidColorBrush.ColorAnimations>
        <!-- Because the AutoReverse property is set to True, the color will animate back to
        Red after it animates to Blue. -->
        <ColorAnimation From="Red" To="Blue" Begin="0" Duration="5"
          AutoReverse="True" RepeatCount="100"/>
      </SolidColorBrush.ColorAnimations>
    </SolidColorBrush>
  </Ellipse.Fill>
</Ellipse>

</Canvas>
```

In the previous example, the ColorAnimationCollection tag, <ColorAnimationCollection>, is omitted when animating the brush's color. When animating a designated animation property—properties of the form PropertyNameAnimations, such as the ColorAnimations property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from UIElement—you must nest the animations within an animation collection tag. For more information about animating properties, see Animation in "Avalon".

This example demonstrates how to use the By, From, and To properties of animations to set an animation's starting and ending values in "Longhorn" markup language (code-named "XAML"). In the following markup, LengthAnimation objects are used to animate the endpoints of five Line elements. Although this example uses the LengthAnimation, the behavior of the From, To, and By properties is the same for all the animation classes.

In the first markup fragment, the X2 attribute of the first line is animated from 50 to 100 over a duration of 10 seconds. Because the From and To properties of the LengthAnimation are set, the animation ignores the line's base value, starting at the specified From value and moving toward the specified To value.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">
```

```

<Line ID="Line1" X1="10" Y1="10" X2="100" Y2="50"
  Stroke="Black" StrokeThickness="5">
  <Line.X2>
    <LengthAnimationCollection>
      <LengthAnimation From="50" To="300" Duration="10" RepeatCount="50" />
    </LengthAnimationCollection>
  </Line.X2>
</Line>

```

The second line's animation has only its To property set. When the From value of an animation isn't set, the animation uses the base value of the property it is animating or the ending value of a previous animation. In this example, the animation uses the base value of the line's X2 property, 100, as its starting value.

```

  <Line ID="Line2" X1="10" Y1="70" X2="100" Y2="70" Stroke="Black" StrokeThickness="5">
  <Line.X2>
    <LengthAnimationCollection>
      <LengthAnimation To="300" Duration="10" RepeatCount="50" />
    </LengthAnimationCollection>
  </Line.X2>
</Line>

```

The third line's animation has only its By property set. The By of an animation specifies "by how much" the animation changes a value over its duration. As in the previous example, the animation uses the base value of the property it is animating or the ending value of a previous animation. In this example, the animation uses the base value of the line's X2 property, 100, as its starting value, and adds 300 to that value over a duration of 10 seconds.

```

  <Line ID="Line3" X1="10" Y1="130" X2="100" Y2="130" Stroke="Black" StrokeThickness="5">
  <Line.X2>
    <LengthAnimationCollection>
      <LengthAnimation By="300" Duration="10" RepeatCount="50" />
    </LengthAnimationCollection>
  </Line.X2>
</Line>

```

The fourth line's animation has its By and From properties set. As a result, the line's X2 attribute is animated from 50 to 350 over a duration of 10 seconds.

```

  <Line ID="Line4" X1="10" Y1="190" X2="100" Y2="190" Stroke="Black" StrokeThickness="5">
  <Line.X2>
    <LengthAnimationCollection>
      <LengthAnimation From="50" By="300" Duration="10" RepeatCount="50" />
    </LengthAnimationCollection>
  </Line.X2>
</Line>

```

The fifth line's animation has only its From value set. When an animation has no explicit destination value, it uses the base value of the property it is animating or the output of a previous animation as its destination value. In this case, the line's X2 attribute is animated from 50 to 100.

```

  <Line ID="Line5" X1="10" Y1="250" X2="100" Y2="250" Stroke="Black" StrokeThickness="5">
  <Line.X2>
    <LengthAnimationCollection>

```

```

        <LengthAnimation From="50" Duration="10" RepeatCount="50" />
    </LengthAnimationCollection>
</Line.X2>
</Line>

</Canvas>

```

This example demonstrates how to animate a property using "XAML". To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the `RectangleWidth` of a `Rectangle`, the `RectangleWidth` must be set to a non-animated value (in this case, a `Length` object).

In the following example, the `Width` of a `Button` is animated. Because the `Width` property takes a `Length`, a `LengthAnimation` is needed. A `LengthAnimationCollection` is used to contain the `LengthAnimation`. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no `WidthAnimations` property, the `LengthAnimationCollection` is associated directly with the button's `Width` property.

```

        <Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

    <Button.Width>
        <LengthAnimationCollection>
            <LengthAnimation To="50" Duration="5" RepeatCount="500"
                AutoReverse="True"/>
        </LengthAnimationCollection>
    </Button.Width>

    A Button
</Button>

```

In the next example, the `Background` color of a second button is animated. The `Background` property takes a `Brush`. In this example, a `SolidColorBrush` is used to fill the button's `Background`, although a gradient, image, or pattern could have been used. To animate the button's background color, the `Color` of the `SolidColorBrush` must be animated. Because the `SolidColorBrush.Color` property accepts a `Color`, a `ColorAnimation` is used to animate the property. The `SolidColorBrush.Color` property has a corresponding `ColorAnimations` property, so the `ColorAnimation` is nested within the `ColorAnimations` property in order to animate the color of the brush.

```

        <Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">

    <Button.Background>
        <SolidColorBrush Color="Blue">
            <SolidColorBrush.ColorAnimations>
                <ColorAnimation From="Red" To="Blue" Duration="7"
                    RepeatCount="500" AutoReverse="True"/>
            </SolidColorBrush.ColorAnimations>
        </SolidColorBrush>
    </Button.Background>

```

```
Another Button
</Button>
```

```
</Canvas>
```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag. For more information about animating properties, see [Animation in "Avalon"](#).

This example demonstrates how to create an animation that repeats indefinitely. To make an animation repeat indefinitely in "XAML", set the animation's `RepeatDuration` property to `Indefinite`. In code, set the animation's `RepeatDuration` property to `Time.Indefinite` or set its `RepeatCount` property to `double.PositiveInfinity`.

In the following examples, a `LengthAnimation` is set to repeat indefinitely. Although this example uses a `LengthAnimation`, the procedure is the same for all the animation classes.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Line X1="10" Y1="20" X2="50" Y2="20"
StrokeThickness="10" Stroke="Black">
  <Line.X2>
    <LengthAnimationCollection>
      <LengthAnimation From="30" To="300" Duration="10"
        RepeatDuration="Indefinite" />
    </LengthAnimationCollection>
  </Line.X2>
</Line>

</Canvas>

// C#
Line myLine = new Line();

LengthAnimation myLengthAnimation = new LengthAnimation();
myLengthAnimation.From = new Length(30);
myLengthAnimation.To = new Length(300);
myLengthAnimation.Duration = new Time(10000);
myLengthAnimation.RepeatDuration = Time.Indefinite;

LengthAnimationCollection collection = new LengthAnimationCollection();
collection.Add(myLengthAnimation);

myLine.SetAnimations(Line.X2Property, collection);

' VB .NET
Dim myLine As new MSAvalon.Windows.Shapes.Line

Dim myLengthAnimation As new MSAvalon.Windows.Media.Animation.LengthAnimation
myLengthAnimation.From = new MSAvalon.Windows.Length(30)
myLengthAnimation.To = new MSAvalon.Windows.Length(300)
```

```
myLengthAnimation.Duration = new MS Avalon.Windows.Media.Animation.Time(10000)
myLengthAnimation.RepeatDuration = _
    MS Avalon.Windows.Media.Animation.Time.Indefinite
```

```
Dim collection As new MS Avalon.Windows.Media.Animation.LengthAnimationCollection
collection.Add(myLengthAnimation)
```

```
myLine.SetAnimations(Line.X2Property, collection)
```

ColorAnimationCollection Class

Definition: Represents a collection of ColorModifier animations.

Method	Description
Add	Adds animations to the collection.
AddChild	Implementation of AddChild. Adds a Modifier to this AnimationCollection from Markup.
AddText	Implementation of AddText. This is not implemented on this class.
Apply	Implementation of Apply. Applies an animation collection in markup to an element.
Clear	Clears the collection by setting the collection's Count to 0.
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorAnimationCollection	Initializes a new instance of the ColorAnimationCollection class.
Contains	Returns a Boolean that indicates whether the collection contains the specified ColorModifier.
Copy	Creates a copy of this ColorAnimationCollection.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	Copies the entire ColorAnimationCollection to the specified one-dimensional array, starting at the specified index of the target array.
Disable	Inherited from AnimationCollection.
DisableImpl	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Inherited from AnimationCollection.
EnableImpl	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator Inherited from AnimationCollection.

GetEnumeratorImpl	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Calculates and returns the output of the animation collection.
GetValueImpl	Provides a Modifier at a given index.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Addition	
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
SetDefaultParentTimeline	Inherited from AnimationCollection.
SetValueImpl	Sets a Modifier at a given index.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AnimationType	Gets the type of animation stored in the collection.

CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Inherited from AnimationCollection.
CountImpl	
Empty	An unchangeable empty ColorAnimationCollection.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Returns true if at least one of the animations in the animation list is currently active. Inherited from AnimationCollection.
IsFixedSize	
IsOverridingBaseValue	Returns true if at least one of the animations in the animation list is currently on. Inherited from AnimationCollection.
IsReadOnly	
IsSynchronized	
IsUsingBaseValue	Inherited from AnimationCollection.
IsUsingBaseValueImpl	
Item	Gets or sets the animation at the specified index.
Item	Use this to get or set a Modifier at a given index. Inherited from AnimationCollection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The AnimationCollection classes, such as ColorAnimationCollection, provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate a base value. When a property requests the current value of an ColorAnimationCollection, the property calls the ColorAnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate a property using "Longhorn" markup language (code-named "XAML"). To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the RectangleWidth of a Rectangle, the RectangleWidth must be set to a non-animated value (in this case, a Length object).

In the following example, the Width of a Button is animated. Because the Width property takes a Length, a LengthAnimation is needed. A LengthAnimationCollection is used to contain the LengthAnimation. Although in this example there is only one animation, you could associate multiple animations with a

single property by placing the animations within the collection. Because there is no `WidthAnimations` property, the `LengthAnimationCollection` is associated directly with the button's `Width` property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

<Button.Width>
  <LengthAnimationCollection>
    <LengthAnimation To="50" Duration="5" RepeatCount="500"
      AutoReverse="True"/>
  </LengthAnimationCollection>
</Button.Width>

  A Button
</Button>
```

In the next example, the `Background` color of a second button is animated. The `Background` property takes a `Brush`. In this example, a `SolidColorBrush` is used to fill the button's `Background`, although a gradient, image, or pattern could have been used. To animate the button's background color, the `Color` of the `SolidColorBrush` must be animated. Because the `SolidColorBrush.Color` property accepts a `Color`, a `ColorAnimation` is used to animate the property. The `SolidColorBrush.Color` property has a corresponding `ColorAnimations` property, so the `ColorAnimation` is nested within the `ColorAnimations` property in order to animate the color of the brush.

```
<Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">

<Button.Background>
  <SolidColorBrush Color="Blue">
    <SolidColorBrush.ColorAnimations>
      <ColorAnimation From="Red" To="Blue" Duration="7"
        RepeatCount="500" AutoReverse="True"/>
    </SolidColorBrush.ColorAnimations>
  </SolidColorBrush>
</Button.Background>

  Another Button
</Button>

</Canvas>
```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

ColorKeyFrameCollection Class

	Method	Description
Add		Strongly typed implementation of <code>Add</code> .

CloneCore	Implementation of CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorKeyFrameCollection	
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy	Creates a new ColorKeyFrameCollection
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetCurrentSegmentValues	TODO
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
Validate	
ValidateObjectState	Implementation of ValidateObjectState.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.

WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.
---------------	--

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Implementation of Count.
Destination	The value specified in the last KeyFrame.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

ColorModifier Class

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorModifier	
Copy	Creates a copy of this ColorModifier
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.

Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	
GetValueImpl	
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a ColorModifier
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.

IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from Modifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from Modifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

ColorTimedModifier Class

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current time.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
ColorTimedModifier	
Copy	Creates a copy of this ColorTimedModifier
Copy	Creates a copy of this ColorModifier Inherited from ColorModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null.
EndIn	Schedules an interactive end time.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	
Finalize	Allows an Object to attempt to free resources and perform other cleanup

	operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Inherited from ColorModifier.
GetValueImpl	Inherited from ColorModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a ColorModifier Inherited from ColorModifier.
Pause	Pauses this timeline.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this timeline.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in

	as a parameter to <code>ChangeableHelper.UseChangeable</code> , to help determine when a <code>Changeable</code> being put into "use" should be promoted to " <code>ChangeableReference</code> ". Inherited from <code>Changeable</code> .
<code>AutoReverse</code>	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration.
<code>Begin</code>	Gets or sets an offset to the start time of the animation.
<code>CanMakeUnchangeable</code>	True if this <code>Changeable</code> can be made unchangeable. Inherited from <code>Changeable</code> .
<code>CurrentRepeat</code>	Gets the number of the current iteration of the animation.
<code>CurrentTime</code>	Gets the current time value of the animation.
<code>Deceleration</code>	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase.
<code>Duration</code>	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation.
<code>End</code>	Gets or sets the maximum end time of the animation.
<code>EndSync</code>	Gets or sets a <code>TimeEndSync</code> enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the <code>Duration</code> property is not explicitly set.
<code>Fill</code>	Gets or sets a value that specifies the state of an object when its animation ends.
<code>FillDefault</code>	Gets or sets a value that indicates the default value of the <code>Fill</code> property of the current animation and its child timelines.
<code>IsChangeable</code>	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from <code>Changeable</code> .
<code>IsChanging</code>	Gets a value that indicates whether the animation is active.
<code>IsEnabled</code>	
<code>IsForwardProgressing</code>	Gets a value that indicates whether the animation is progressing from past to future.
<code>IsOverridingBaseValue</code>	Gets a value that indicates whether the animation is active or in a fill period.
<code>IsPaused</code>	Gets a value that indicates whether the animation is active and paused.
<code>IsReversed</code>	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline.
<code>ParentTimeline</code>	Gets or sets the default parent timeline of the animation.
<code>Progress</code>	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed.
<code>RepeatCount</code>	Gets or sets the number of times an animation should repeat.
<code>RepeatDuration</code>	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, it will repeat itself for the length of time specified by this property.
<code>Restart</code>	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached.
<code>RestartDefault</code>	Gets or sets the default value of the <code>Restart</code> property of the current animation and its child timelines.
<code>Speed</code>	Gets or sets the relative speed at which time should pass for the animation, compared to its parent timeline.

StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

DecimalAnimationCollection Class

Definitions: Represents a collection of DecimalModifier animations.

Method	Description
Add	The Add(DecimalModifier) and Add(Object) methods add animations to the collection; the Add(Decimal,DecimalAnimationCollection) method calculates the current value of the specified collection based on the specified base value.
AddChild	Implementation of AddChild. Adds a Modifier to this AnimationCollection from Markup.
AddText	Implementation of AddText. This is not implemented on this class.
Apply	Implementation of Apply. Applies an animation collection in markup to an element.
Clear	Clears the collection by setting the collection's Count to 0.
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a Boolean that indicates whether the collection contains the specified DecimalModifier.
Copy	Creates a copy of this DecimalAnimationCollection.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	Copies the entire DecimalAnimationCollection to the specified one-dimensional array, starting at the specified index of the target array.
DecimalAnimationCollection	Creates an empty DecimalAnimationCollection with a default capacity for a single animation.
Disable	Inherited from AnimationCollection.
DisableImpl	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Inherited from AnimationCollection.
EnableImpl	

Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator Inherited from AnimationCollection.
GetEnumeratorImpl	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Returns the current value of the animation.
GetValueImpl	Provides a Modifier at a given index.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Addition	
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
SetDefaultParentTimeline	Inherited from AnimationCollection.
SetValueImpl	Sets a Modifier at a given index.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AnimationType	AnimationType
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Inherited from AnimationCollection.
CountImpl	
Empty	An unchangeable empty DecimalAnimationCollection.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Returns true if at least one of the animations in the animation list is currently active. Inherited from AnimationCollection.
IsFixedSize	
IsOverridingBaseValue	Returns true if at least one of the animations in the animation list is currently on. Inherited from AnimationCollection.
IsReadOnly	
IsSynchronized	
IsUsingBaseValue	Inherited from AnimationCollection.
IsUsingBaseValueImpl	
Item	this - typed version of indexer
Item	Use this to get or set a Modifier at a given index. Inherited from AnimationCollection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The AnimationCollection classes, such as DecimalAnimationCollection, provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate a base value. When a property requests the current value of an DecimalAnimationCollection, the property calls the DecimalAnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate a property using "Longhorn" markup language (code-named "XAML"). To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the `RectangleWidth` of a `Rectangle`, the `RectangleWidth` must be set to a non-animated value (in this case, a `Length` object).

In the following example, the `Width` of a `Button` is animated. Because the `Width` property takes a `Length`, a `LengthAnimation` is needed. A `LengthAnimationCollection` is used to contain the `LengthAnimation`. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no `WidthAnimations` property, the `LengthAnimationCollection` is associated directly with the button's `Width` property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

<Button.Width>
  <LengthAnimationCollection>
    <LengthAnimation To="50" Duration="5" RepeatCount="500"
      AutoReverse="True"/>
  </LengthAnimationCollection>
</Button.Width>

  A Button
</Button>
```

In the next example, the `Background` color of a second button is animated. The `Background` property takes a `Brush`. In this example, a `SolidColorBrush` is used to fill the button's `Background`, although a gradient, image, or pattern could have been used. To animate the button's background color, the `Color` of the `SolidColorBrush` must be animated. Because the `SolidColorBrush.Color` property accepts a `Color`, a `ColorAnimation` is used to animate the property. The `SolidColorBrush.Color` property has a corresponding `ColorAnimations` property, so the `ColorAnimation` is nested within the `ColorAnimations` property in order to animate the color of the brush.

```
  <Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">

<Button.Background>
  <SolidColorBrush Color="Blue">
    <SolidColorBrush.ColorAnimations>
      <ColorAnimation From="Red" To="Blue" Duration="7"
        RepeatCount="500" AutoReverse="True"/>
    </SolidColorBrush.ColorAnimations>
  </SolidColorBrush>
</Button.Background>

  Another Button
</Button>

</Canvas>
```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit

the animation collection tag. However, when animating a property of a UI element—those classes that derive from UIElement—you must nest the animations within an animation collection tag.

DecimalModifier Class

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this DecimalModifier
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DecimalModifier	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	
GetValueImpl	
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a DecimalModifier
PropagateEventHandler	Shares a Changed event handler with the current object's data

	members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from Modifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from Modifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

DecimalTimedModifier Class

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current time.
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.

CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this DecimalModifier Inherited from DecimalModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
Copy DecimalTimedModifier	Creates a copy of this DecimalTimedModifier
Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null.
EndIn	Schedules an interactive end time.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Inherited from DecimalModifier.
GetValueImpl	Inherited from DecimalModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a DecimalModifier Inherited from DecimalModifier.
Pause	Pauses this timeline.

PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this timeline.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration.
Begin	Gets or sets an offset to the start time of the animation.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	Gets the number of the current iteration of the animation.
CurrentTime	Gets the current time value of the animation.
Deceleration	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase.
Duration	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation.
End	Gets or sets the maximum end time of the animation.
EndSync	Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set.
Fill	Gets or sets a value that specifies the state of an object when its animation ends.
FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines.
IsChangeable	Gets a Boolean that indicates whether the object is currently

	modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active.
IsEnabled	
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period.
IsPaused	Gets a value that indicates whether the animation is active and paused.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline.
ParentTimeline	Gets or sets the default parent timeline of the animation.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed.
RepeatCount	Gets or sets the number of times an animation should repeat.
RepeatDuration	
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached.
RestartDefault	Gets or sets the default value of the Restart property of the current animation and its child timelines.
Speed	Gets or sets the relative speed at which time should pass for the animation, compared to its parent timeline.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

DoubleAnimation Class

Method	Description
BeginIn	Starts or restarts the animation at the specified offset from the current time. Inherited from DoubleTimedModifier.
CloneCore	Implementation of CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this DoubleTimedModifier Inherited from DoubleTimedModifier.
Copy	Creates a copy of this DoubleModifier Inherited from DoubleModifier.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.

Disable	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to Enable. Inherited from DoubleTimedModifier.
DoubleAnimation	Initializes a new instance of the DoubleAnimation class.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Enables this timeline, parenting it to the timeline specified by the ParentTimeline property. This allows the timeline to become active. This method throws an exception if the ParentTimeline property is null. Inherited from DoubleTimedModifier.
EndIn	Schedules an interactive end time. Inherited from DoubleTimedModifier.
Equals	Determines whether two Object instances are equal. Inherited from Object.
FillInClone	Inherited from DoubleTimedModifier.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Calculates the value of the animation at the current time.
GetValueImpl	Inherited from DoubleModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a DoubleModifier Inherited from DoubleModifier.
Pause	Pauses this timeline. Inherited from DoubleTimedModifier.
PropagateEventHandler	PropagateEventHandler
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.

Resume	Resumes this timeline. Inherited from DoubleTimedModifier.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time. Inherited from DoubleTimedModifier.
SetDefaultParentTimeline	
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase. Inherited from DoubleTimedModifier.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration. Inherited from DoubleTimedModifier.
Begin	Gets or sets an offset to the start time of the animation. Inherited from DoubleTimedModifier.
By	Gets or sets the total amount by which the animation changes its starting value.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	Gets the number of the current iteration of the animation. Inherited from DoubleTimedModifier.
CurrentTime	Gets the current time value of the animation. Inherited from DoubleTimedModifier.
Deceleration	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase. Inherited from DoubleTimedModifier.
Duration	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation. Inherited from DoubleTimedModifier.
End	Gets or sets the maximum end time of the animation. Inherited from DoubleTimedModifier.
EndSync	Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set. Inherited from DoubleTimedModifier.
Fill	Gets or sets a value that specifies the state of an object when its animation ends. Inherited from DoubleTimedModifier.

FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines. Inherited from DoubleTimedModifier.
From	Gets or sets the starting value of an animation.
InterpolationMethod	InterpolationMethod
IsAdditive	IsAdditive
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from DoubleTimedModifier.
IsCumulative	IsCumulative
IsEnabled	Inherited from DoubleTimedModifier.
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future. Inherited from DoubleTimedModifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from DoubleTimedModifier.
IsPaused	Gets a value that indicates whether the animation is active and paused. Inherited from DoubleTimedModifier.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline. Inherited from DoubleTimedModifier.
KeyFrames	KeyValues
ParentTimeline	Gets or sets the default parent timeline of the animation. Inherited from DoubleTimedModifier.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed. Inherited from DoubleTimedModifier.
RepeatCount	Gets or sets the number of times an animation should repeat. Inherited from DoubleTimedModifier.
RepeatDuration	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, it will repeat itself for the length of time specified by this property. Inherited from DoubleTimedModifier.
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached. Inherited from DoubleTimedModifier.
RestartDefault	Gets or sets the default value of the Restart property of the current animation and its child timelines. Inherited from DoubleTimedModifier.
Speed	Gets or sets the relative speed at which time should pass for the animation, compared to its parent timeline. Inherited from DoubleTimedModifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
Timeline	Timeline Inherited from DoubleTimedModifier.
To	Gets or sets the ending value of the animation.
UIContext	Gets the UIContext of the current object. The UIContext is used for

maintaining thread safety. Inherited from `Changeable`.

`UsesBaseValue`

This example demonstrates how to animate the size of an element using "Longhorn" markup language (code-named "XAML"). There are multiple ways to animate the size of an element: directly animate the height and width attributes of the element, or apply an animated `ScaleTransform` to the element. In this example, two `Rectangle` elements are resized using these methods. One rectangle is resized by animating its `RectangleWidth` attribute and another is resized by animating a `ScaleTransform` applied to the rectangle. Each rectangle is filled with a pattern to highlight the differences between the two resizing methods. Initially, the two patterns look the same, but as the rectangles are resized, patterns change depending on how their containing rectangle is resized.

In the first example, a `Rectangle` element's `RectangleWidth` property is animated using a `LengthAnimationCollection` and a `LengthAnimation`. The `LengthAnimation` object in this example animates the rectangle's `RectangleWidth` from its base value of 200 To a destination value of 600 over a Duration of 10 seconds.

```
<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Rectangle
    RectangleTop="20"
    RectangleLeft="20"
    RectangleWidth="200"
    RectangleHeight="150"
    Stroke="Red"
    StrokeThickness="5">

    <Rectangle.Fill>
      <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
        ImageSource="help.gif" TileMode="Tile"/>
    </Rectangle.Fill>

    <Rectangle.RectangleWidth>
      <!-- Animate the Rectangle's width: -->
      <LengthAnimationCollection>
        <LengthAnimation
          To="600" Duration="10" AutoReverse="true" RepeatCount="50" />
      </LengthAnimationCollection>
    </Rectangle.RectangleWidth>

  </Rectangle>
```

When the previous "XAML" is run, more of the pattern is exposed as the rectangle expands; however, the question marks that make up the pattern do not grow larger.

In the next example, a `TransformDecorator` is used to apply a `ScaleTransform` to a rectangle. A `DoubleAnimation` is used to animate the `ScaleTransform` object's `ScaleX` value using the `ScaleXAnimations` attribute. The `DoubleAnimation` animates the `ScaleX` value From 1 To a destination value of 3 over a Duration of 10 seconds. As a result, the rectangle's width is scaled from 100 percent (its starting size) to 300 percent over ten seconds.

```
<TransformDecorator AffectsLayout="False">
  <TransformDecorator.Transform>
```

```

<!-- Use the ScaleTransform to enlarge the rectangle -->
<ScaleTransform ScaleX="1" ScaleY="1">
  <ScaleTransform.ScaleXAnimations>
    <DoubleAnimation From="1" To="3" RepeatCount="30"
      AutoReverse="True" Begin="0" Duration="10" />
  </ScaleTransform.ScaleXAnimations>
</ScaleTransform>

</TransformDecorator.Transform>

<Rectangle
  RectangleLeft="20"
  RectangleTop="200"
  RectangleWidth="200"
  RectangleHeight="150"
  Stroke="Black"
  StrokeThickness="3">
  <Rectangle.Fill>
    <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
      ImageSource="help.gif" TileMode="Tile"/>
  </Rectangle.Fill>
</Rectangle>
</TransformDecorator>

</Canvas>

```

In the previous example, the `DoubleAnimationCollection` tag, `<DoubleAnimationCollection>`, is omitted when animating the transformation's scale factor. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ScaleXAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

When the second rectangle expands, the objects in the pattern also grow larger, unlike in the first rectangle. The pattern behaves this way because when you transform an element the entire element and its child elements are transformed. When you directly alter the size of an element, as in the case of the first rectangle, the element's children are not resized, unless their size and position are dependent on the size of their parent element.

This example demonstrates how to use the `By`, `From`, and `To` properties of animations to set an animation's starting and ending values in "XAML". In the following markup, `LengthAnimation` objects are used to animate the endpoints of five `Line` elements. Although this example uses the `LengthAnimation`, the behavior of the `From`, `To`, and `By` properties is the same for all the animation classes. In the first markup fragment, the `X2` attribute of the first line is animated from 50 to 100 over a duration of 10 seconds. Because the `From` and `To` properties of the `LengthAnimation` are set, the animation ignores the line's base value, starting at the specified `From` value and moving toward the specified `To` value.

```

<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Line ID="Line1" X1="10" Y1="10" X2="100" Y2="50"
    Stroke="Black" StrokeThickness="5">
    <Line.X2>
      <LengthAnimationCollection>
        <LengthAnimation From="50" To="300" Duration="10" RepeatCount="50" />
      </LengthAnimationCollection>
    </Line.X2>
  </Line>

```

```

</Line.X2>
</Line>

```

The second line's animation has only its To property set. When the From value of an animation isn't set, the animation uses the base value of the property it is animating or the ending value of a previous animation. In this example, the animation uses the base value of the line's X2 property, 100, as its starting value.

```

<Line ID="Line2" X1="10" Y1="70" X2="100" Y2="70" Stroke="Black" StrokeThickness="5">
<Line.X2>
  <LengthAnimationCollection>
    <LengthAnimation To="300" Duration="10" RepeatCount="50" />
  </LengthAnimationCollection>
</Line.X2>
</Line>

```

The third line's animation has only its By property set. The By of an animation specifies "by how much" the animation changes a value over its duration. As in the previous example, the animation uses the base value of the property it is animating or the ending value of a previous animation. In this example, the animation uses the base value of the line's X2 property, 100, as its starting value, and adds 300 to that value over a duration of 10 seconds.

```

<Line ID="Line3" X1="10" Y1="130" X2="100" Y2="130" Stroke="Black" StrokeThickness="5">
<Line.X2>
  <LengthAnimationCollection>
    <LengthAnimation By="300" Duration="10" RepeatCount="50" />
  </LengthAnimationCollection>
</Line.X2>
</Line>

```

The fourth line's animation has its By and From properties set. As a result, the line's X2 attribute is animated from 50 to 350 over a duration of 10 seconds.

```

<Line ID="Line4" X1="10" Y1="190" X2="100" Y2="190" Stroke="Black" StrokeThickness="5">
<Line.X2>
  <LengthAnimationCollection>
    <LengthAnimation From="50" By="300" Duration="10" RepeatCount="50" />
  </LengthAnimationCollection>
</Line.X2>
</Line>

```

The fifth line's animation has only its From value set. When an animation has no explicit destination value, it uses the base value of the property it is animating or the output of a previous animation as its destination value. In this case, the line's X2 attribute is animated from 50 to 100.

```

<Line ID="Line5" X1="10" Y1="250" X2="100" Y2="250" Stroke="Black" StrokeThickness="5">
<Line.X2>
  <LengthAnimationCollection>
    <LengthAnimation From="50" Duration="10" RepeatCount="50" />
  </LengthAnimationCollection>
</Line.X2>
</Line>

```

</Canvas>

This example demonstrates how to create an animation that repeats indefinitely. To make an animation repeat indefinitely in "XAML", set the animation's RepeatDuration property to Indefinite. In code, set the animation's RepeatDuration property to Time.Indefinite or set its RepeatCount property to double.PositiveInfinity.

In the following examples, a LengthAnimation is set to repeat indefinitely. Although this example uses a LengthAnimation, the procedure is the same for all the animation classes.

```
<Canvas ID="root"

xmlns="http://schemas.microsoft.com/2003/xaml">

<Line X1="10" Y1="20" X2="50" Y2="20"
StrokeThickness="10" Stroke="Black">
  <Line.X2>
    <LengthAnimationCollection>
      <LengthAnimation From="30" To="300" Duration="10"
        RepeatDuration="Indefinite" />
    </LengthAnimationCollection>
  </Line.X2>
</Line>

</Canvas>

// C#
Line myLine = new Line();

LengthAnimation myLengthAnimation = new LengthAnimation();
myLengthAnimation.From = new Length(30);
myLengthAnimation.To = new Length(300);
myLengthAnimation.Duration = new Time(10000);
myLengthAnimation.RepeatDuration = Time.Indefinite;

LengthAnimationCollection collection = new LengthAnimationCollection();
collection.Add(myLengthAnimation);

myLine.SetAnimations(Line.X2Property, collection);

' VB .NET
Dim myLine As new MSAvalon.Windows.Shapes.Line

Dim myLengthAnimation As new MSAvalon.Windows.Media.Animation.LengthAnimation
myLengthAnimation.From = new MSAvalon.Windows.Length(30)
myLengthAnimation.To = new MSAvalon.Windows.Length(300)
myLengthAnimation.Duration = new MSAvalon.Windows.Media.Animation.Time(10000)
myLengthAnimation.RepeatDuration = _
    MSAvalon.Windows.Media.Animation.Time.Indefinite

Dim collection As new MSAvalon.Windows.Media.Animation.LengthAnimationCollection
collection.Add(myLengthAnimation)

myLine.SetAnimations(Line.X2Property, collection)
```

DoubleAnimationCollection Class

Definition: Represents a collection of DoubleModifier animations.

Method	Description
Add	The Add(DoubleModifier) and Add(Object) methods add animations to the collection. The Add(Double, DoubleAnimationCollection) method calculates the current value of the specified collection based on the specified base value.
AddChild	Implementation of AddChild. Adds a Modifier to this AnimationCollection from Markup.
AddText	Implementation of AddText. This is not implemented on this class.
Apply	Implementation of Apply. Applies an animation collection in markup to an element.
Clear	Clears the collection by setting the collection's Count to 0.
CloneCore	CloneCore
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Contains	Returns a Boolean that indicates whether the collection contains the specified DoubleModifier.
Copy	Creates a copy of this DoubleAnimationCollection.
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
CopyTo	Copies the entire DoubleAnimationCollection to the specified one-dimensional array, starting at the specified index of the target array.
Disable	Inherited from AnimationCollection.
DisableImpl	
DoubleAnimationCollection	Creates an empty DoubleAnimationCollection with a default capacity for a single animation.
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Enable	Inherited from AnimationCollection.
EnableImpl	
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetEnumerator	GetEnumerator Inherited from AnimationCollection.
GetEnumeratorImpl	Returns an object that can be used to enumerate items in the list.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.

GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Returns the current value of the animation.
GetValueImpl	Provides a Modifier at a given index.
IndexOf	
Insert	
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	Implementation of MakeUnchangeableCore.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	OnChanged
op_Addition	
op_Implicit	
PropagateEventHandler	Implementation of PropagateEventHandler.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Remove	
RemoveAt	
SetDefaultParentTimeline	Inherited from AnimationCollection.
SetValueImpl	Sets a Modifier at a given index.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AnimationType	AnimationType
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Inherited from AnimationCollection.
CountImpl	
Empty	An unchangeable empty DoubleAnimationCollection.

IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Returns true if at least one of the animations in the animation list is currently active. Inherited from AnimationCollection.
IsFixedSize	
IsOverridingBaseValue	Returns true if at least one of the animations in the animation list is currently on. Inherited from AnimationCollection.
IsReadOnly	
IsSynchronized	
IsUsingBaseValue	Inherited from AnimationCollection.
IsUsingBaseValueImpl	
Item	this - typed version of indexer
Item	Use this to get or set a Modifier at a given index. Inherited from AnimationCollection.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
SyncRoot	
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

The AnimationCollection classes, such as DoubleAnimationCollection, provide you with more control over how you animate a property than a single animation could. Each of the AnimationCollection classes effectively functions as a single composite animation, using all the animations in the collection to animate a base value. When a property requests the current value of a DoubleAnimationCollection, the property calls the DoubleAnimationCollection's GetValue method and passes it the property's base value. The first animation in the collection processes this base value and produces a result, which is then passed to the next animation in the collection, and so on, until the value has been processed by all the animations in the collection.

This example demonstrates how to animate the size of an element using "Longhorn" markup language (code-named "XAML"). There are multiple ways to animate the size of an element: directly animate the height and width attributes of the element, or apply an animated ScaleTransform to the element. In this example, two Rectangle elements are resized using these methods. One rectangle is resized by animating its RectangleWidth attribute and another is resized by animating a ScaleTransform applied to the rectangle. Each rectangle is filled with a pattern to highlight the differences between the two resizing methods. Initially, the two patterns look the same, but as the rectangles are resized, patterns change depending on how their containing rectangle is resized.

In the first example, a Rectangle element's RectangleWidth property is animated using a LengthAnimationCollection and a LengthAnimation. The LengthAnimation object in this example animates the rectangle's RectangleWidth from its base value of 200 To a destination value of 600 over a Duration of 10 seconds.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Rectangle
RectangleTop="20"
```

```

RectangleLeft="20"
RectangleWidth="200"
RectangleHeight="150"
Stroke="Red"
StrokeThickness="5">

<Rectangle.Fill>
  <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
    ImageSource="help.gif" TileMode="Tile"/>
</Rectangle.Fill>

<Rectangle.RectangleWidth>
<!-- Animate the Rectangle's width: -->
  <LengthAnimationCollection>
    <LengthAnimation
      To="600" Duration="10" AutoReverse="true" RepeatCount="50" />
  </LengthAnimationCollection>
</Rectangle.RectangleWidth>

</Rectangle>

```

When the previous "XAML" is run, more of the pattern is exposed as the rectangle expands; however, the question marks that make up the pattern do not grow larger.

In the next example, a TransformDecorator is used to apply a ScaleTransform to a rectangle. A DoubleAnimation is used to animate the ScaleTransform object's ScaleX value using the ScaleXAnimations attribute. The DoubleAnimation animates the ScaleX value From 1 To a destination value of 3 over a Duration of 10 seconds. As a result, the rectangle's width is scaled from 100 percent (its starting size) to 300 percent over ten seconds.

```

<TransformDecorator AffectsLayout="False">
  <TransformDecorator.Transform>

    <!-- Use the ScaleTransform to enlarge the rectangle -->
    <ScaleTransform ScaleX="1" ScaleY="1">
      <ScaleTransform.ScaleXAnimations>
        <DoubleAnimation From="1" To="3" RepeatCount="30"
          AutoReverse="True" Begin="0" Duration="10" />
      </ScaleTransform.ScaleXAnimations>
    </ScaleTransform>

  </TransformDecorator.Transform>

  <Rectangle
    RectangleLeft="20"
    RectangleTop="200"
    RectangleWidth="200"
    RectangleHeight="150"
    Stroke="Black"
    StrokeThickness="3">
    <Rectangle.Fill>
      <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
        ImageSource="help.gif" TileMode="Tile"/>
    </Rectangle.Fill>
  </Rectangle>
</TransformDecorator>

```

</Canvas>

In the previous example, the DoubleAnimationCollection tag, <DoubleAnimationCollection>, is omitted when animating the transformation's scale factor. When animating a designated animation property—properties of the form PropertyNameAnimations, such as the ScaleXAnimations property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from UIElement—you must nest the animations within an animation collection tag.

When the second rectangle expands, the objects in the pattern also grow larger, unlike in the first rectangle. The pattern behaves this way because when you transform an element the entire element and its child elements are transformed. When you directly alter the size of an element, as in the case of the first rectangle, the element's children are not resized, unless their size and position are dependent on the size of their parent element.

This example demonstrates how to animate a property using "XAML". To animate a property, you associate the proper animation collection and animations with the property either directly or using the property's corresponding animation property. There are a variety of animation classes, each of which animates a different kind of value.

Before some properties can be animated, they must be given a base value. For example, before animating the RectangleWidth of a Rectangle, the RectangleWidth must be set to a non-animated value (in this case, a Length object).

In the following example, the Width of a Button is animated. Because the Width property takes a Length, a LengthAnimation is needed. A LengthAnimationCollection is used to contain the LengthAnimation. Although in this example there is only one animation, you could associate multiple animations with a single property by placing the animations within the collection. Because there is no WidthAnimations property, the LengthAnimationCollection is associated directly with the button's Width property.

```
<Canvas ID="root"
xmlns="http://schemas.microsoft.com/2003/xaml">

<Button Canvas.Top="20" Canvas.Left="20"
Height="30" Width="200">

  <Button.Width>
    <LengthAnimationCollection>
      <LengthAnimation To="50" Duration="5" RepeatCount="500"
        AutoReverse="True"/>
    </LengthAnimationCollection>
  </Button.Width>

  A Button
</Button>
```

In the next example, the Background color of a second button is animated. The Background property takes a Brush. In this example, a SolidColorBrush is used to fill the button's Background, although a gradient, image, or pattern could have been used. To animate the button's background color, the Color of the SolidColorBrush must be animated. Because the SolidColorBrush.Color property accepts a Color, a ColorAnimation is used to animate the property. The SolidColorBrush.Color property has a corresponding ColorAnimations property, so the ColorAnimation is nested within the ColorAnimations property in order to animate the color of the brush.

```
<Button Canvas.Top="70" Canvas.Left="20"
Height="30" Width="200">
```

```

<Button.Background>
  <SolidColorBrush Color="Blue">
    <SolidColorBrush.ColorAnimations>
      <ColorAnimation From="Red" To="Blue" Duration="7"
        RepeatCount="500" AutoReverse="True"/>
    </SolidColorBrush.ColorAnimations>
  </SolidColorBrush>
</Button.Background>

```

```

Another Button
</Button>

```

```

</Canvas>

```

In the previous example, the `ColorAnimationCollection` tag, `<ColorAnimationCollection>`, is omitted when animating the brush's color. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ColorAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

DoubleKeyFrameCollection Class

Method	Description
Add	Strongly typed implementation of Add.
CloneCore	Implementation of CloneCore.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .
Copy	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
Copy	Creates a new <code>DoubleKeyFrameCollection</code>
DoubleKeyFrameCollection	
EmbeddedChangeableReader	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
EmbeddedChangeableWriter	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
Equals	Determines whether two <code>Object</code> instances are equal. Inherited from <code>Object</code> .
Finalize	Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. Inherited from <code>Object</code> .
GetCurrentSegmentValues	TODO
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from <code>Object</code> .
GetType	Gets the <code>Type</code> of the current instance. Inherited from <code>Object</code> .
MakeUnchangeable	Makes an object immutable; after this method is called on a <code>Changeable</code> , its <code>IsChangeable</code> property is false. Inherited from

	Changeable.
MakeUnchangeableCore	Makes a Changeable object immutable. Inherited from Changeable.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
ToString	Returns a String that represents the current Object. Inherited from Object.
Validate	
ValidateObjectState	Implementation of ValidateObjectState.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
Count	Implementation of Count.
Destination	The value specified in the last KeyFrame.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
Item	
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.

DoubleModifier Class

Method	Description
CloneCore	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from Changeable. Inherited from Changeable.
CloneDownToUnchangeable	Returns an immutable copy of the specified object. Inherited from Changeable.
Copy	Creates a copy of this DoubleModifier
Copy	Returns a modifiable copy of the current object. The copy's IsChangeable property is true and its StatusOfNextUse is Unchangeable. Inherited from Changeable.
DoubleModifier	
EmbeddedChangeableReader	Accesses the specified Changeable data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from Changeable call this method on data members before they can be retrieved through property calls. Inherited from Changeable.
EmbeddedChangeableWriter	Processes a modified Changeable data member and returns a reference to the processed object. Inherited from Changeable.
Equals	Determines whether two Object instances are equal. Inherited from Object.
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	
GetValueImpl	
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a DoubleModifier
PropagateEventHandler	Shares a Changed event handler with the current object's data members or removes it. Inherited from Changeable.
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple

	members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.

Property	Description
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active. Inherited from Modifier.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period. Inherited from Modifier.
StatusOfNextUse	Gets or sets a UseStatus enumeration that specifies how the Changeable object behaves when it is "used." A Changeable object is considered used in the following situations: the object is set into a Property System property, the object is used as a sub-object in a complex Changeable object, or the object is used in a DrawingContext command. Inherited from Changeable.
UIContext	Gets the UIContext of the current object. The UIContext is used for maintaining thread safety. Inherited from Changeable.
UsesBaseValue	UsesBaseValue Inherited from Modifier.

This example demonstrates how to animate the size of an element using "Longhorn" markup language (code-named "XAML"). There are multiple ways to animate the size of an element: directly animate the height and width attributes of the element, or apply an animated ScaleTransform to the element. In this example, two Rectangle elements are resized using these methods. One rectangle is resized by animating its RectangleWidth attribute and another is resized by animating a ScaleTransform applied to the rectangle. Each rectangle is filled with a pattern to highlight the differences between the two resizing methods. Initially, the two patterns look the same, but as the rectangles are resized, patterns change depending on how their containing rectangle is resized.

In the first example, a Rectangle element's RectangleWidth property is animated using a LengthAnimationCollection and a LengthAnimation. The LengthAnimation object in this example animates the rectangle's RectangleWidth from its base value of 200 To a destination value of 600 over a Duration of 10 seconds.


```

<Canvas ID="root"
  xmlns="http://schemas.microsoft.com/2003/xaml">

  <Rectangle
    RectangleTop="20"
    RectangleLeft="20"
    RectangleWidth="200"
    RectangleHeight="150"
    Stroke="Red"
    StrokeThickness="5">

    <Rectangle.Fill>
      <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
        ImageSource="help.gif" TileMode="Tile"/>
    </Rectangle.Fill>

    <Rectangle.RectangleWidth>
      <!-- Animate the Rectangle's width: -->
      <LengthAnimationCollection>
        <LengthAnimation
          To="600" Duration="10" AutoReverse="true" RepeatCount="50" />
      </LengthAnimationCollection>
    </Rectangle.RectangleWidth>

  </Rectangle>

```

When the previous "XAML" is run, more of the pattern is exposed as the rectangle expands; however, the question marks that make up the pattern do not grow larger.

In the next example, a `TransformDecorator` is used to apply a `ScaleTransform` to a rectangle. A `DoubleAnimation` is used to animate the `ScaleTransform` object's `ScaleX` value using the `ScaleXAnimations` attribute. The `DoubleAnimation` animates the `ScaleX` value From 1 To a destination value of 3 over a Duration of 10 seconds. As a result, the rectangle's width is scaled from 100 percent (its starting size) to 300 percent over ten seconds.

```

<TransformDecorator AffectsLayout="False">
  <TransformDecorator.Transform>

    <!-- Use the ScaleTransform to enlarge the rectangle -->
    <ScaleTransform ScaleX="1" ScaleY="1">
      <ScaleTransform.ScaleXAnimations>
        <DoubleAnimation From="1" To="3" RepeatCount="30"
          AutoReverse="True" Begin="0" Duration="10" />
      </ScaleTransform.ScaleXAnimations>
    </ScaleTransform>

  </TransformDecorator.Transform>

  <Rectangle
    RectangleLeft="20"
    RectangleTop="200"
    RectangleWidth="200"
    RectangleHeight="150"
    Stroke="Black"
    StrokeThickness="3">
    <Rectangle.Fill>

```

```

        <ImageBrush ViewPort="0,0 100,100" ViewPortUnits="Absolute"
        ImageSource="help.gif" TileMode="Tile"/>
    </Rectangle.Fill>
</Rectangle>
</TransformDecorator>

</Canvas>

```

In the previous example, the `DoubleAnimationCollection` tag, `<DoubleAnimationCollection>`, is omitted when animating the transformation's scale factor. When animating a designated animation property—properties of the form `PropertyNameAnimations`, such as the `ScaleXAnimations` property in the previous example—you may omit the animation collection tag. However, when animating a property of a UI element—those classes that derive from `UIElement`—you must nest the animations within an animation collection tag.

When the second rectangle expands, the objects in the pattern also grow larger, unlike in the first rectangle. The pattern behaves this way because when you transform an element the entire element and its child elements are transformed. When you directly alter the size of an element, as in the case of the first rectangle, the element's children are not resized, unless their size and position are dependent on the size of their parent element.

DoubleTimedModifier Class

Method	Description
<code>BeginIn</code>	Starts or restarts the animation at the specified offset from the current time.
<code>CloneCore</code>	Returns a modifiable shallow or deep clone of the current object. This abstract method must be implemented by classes that derive from <code>Changeable</code> . Inherited from <code>Changeable</code> .
<code>CloneDownToUnchangeable</code>	Returns an immutable copy of the specified object. Inherited from <code>Changeable</code> .
<code>Copy</code>	Creates a copy of this <code>DoubleTimedModifier</code>
<code>Copy</code>	Returns a modifiable copy of the current object. The copy's <code>IsChangeable</code> property is true and its <code>StatusOfNextUse</code> is <code>Unchangeable</code> . Inherited from <code>Changeable</code> .
<code>Copy</code>	Creates a copy of this <code>DoubleModifier</code> Inherited from <code>DoubleModifier</code> .
<code>Disable</code>	Disables this timeline, after which the timeline can no longer become active. The timeline can be re-enabled with a call to <code>Enable</code> .
<code>DoubleTimedModifier</code>	
<code>EmbeddedChangeableReader</code>	Accesses the specified <code>Changeable</code> data member, processes it, and returns a reference to the member. This reference should then be reassigned to the original member variable. Classes that derive from <code>Changeable</code> call this method on data members before they can be retrieved through property calls. Inherited from <code>Changeable</code> .
<code>EmbeddedChangeableWriter</code>	Processes a modified <code>Changeable</code> data member and returns a reference to the processed object. Inherited from <code>Changeable</code> .
<code>Enable</code>	Enables this timeline, parenting it to the timeline specified by the <code>ParentTimeline</code> property. This allows the timeline to become active. This method throws an exception if the <code>ParentTimeline</code> property is null.
<code>EndIn</code>	Schedules an interactive end time.
<code>Equals</code>	Determines whether two Object instances are equal. Inherited from

	Object.
FillInClone	
Finalize	Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. Inherited from Object.
GetHashCode	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table. Inherited from Object.
GetType	Gets the Type of the current instance. Inherited from Object.
GetValue	Inherited from DoubleModifier.
GetValueImpl	Inherited from DoubleModifier.
IModifier.GetValue	Inherited from Modifier.
MakeUnchangeable	Makes an object immutable; after this method is called on a Changeable, its IsChangeable property is false. Inherited from Changeable.
MakeUnchangeableCore	MakeUnchangeableCore Inherited from Modifier.
MemberwiseClone	Creates a shallow copy of the current Object. Inherited from Object.
ModifyHandlerIfChangeable	Adds or removes a Changed event handler to or from the specified Changeable object, if the object is currently modifiable. If the specified object is not modifiable—if its IsChangeable property is false—this method has no effect. Inherited from Changeable.
OnChanged	Called when the current object is modified. Classes that derive from Changed should call this method after they have been modified. Inherited from Changeable.
op_Implicit	Implicitly creates an AnimationCollection from a DoubleModifier Inherited from DoubleModifier.
Pause	Pauses this timeline.
PropagateEventHandler	
ReadPreamble	Ensures that simple (non-Changeable) members are being accessed from a valid UI context. This method should be called before any simple members are accessed. Inherited from Changeable.
ReferenceEquals	Determines whether the specified Object instances are the same instance. Inherited from Object.
Resume	Resumes this timeline.
Seek	Moves the current position of the animation backwards or forwards from either the current time, the Begin time, or the End time.
SetDefaultParentTimeline	SetDefaultParentTimeline Inherited from Modifier.
ToString	Returns a String that represents the current Object. Inherited from Object.
ValidateObjectState	Verifies that the current object has a valid state. If the object is in an invalid state, this method throws an exception. Inherited from Changeable.
WritePostscript	Causes the current object to validate itself and then invokes the OnChanged method. Inherited from Changeable.
WritePreamble	Ensures that simple (non-Changeable) members are being accessed from a valid user interface (UI) context. This method should be called before any simple members are set. Inherited from Changeable.
Pr erty	Descripti n

Acceleration	Gets or sets the fraction of the simple duration spent in the acceleration phase.
AllowChangeableReferenceOverride	Used in conjunction with the ChangeableUsageOverride type sent in as a parameter to ChangeableHelper.UseChangeable, to help determine when a Changeable being put into "use" should be promoted to "ChangeableReference". Inherited from Changeable.
AutoReverse	Gets or sets a value that indicates whether the animation plays in reverse after it completes its forward iteration.
Begin	Gets or sets an offset to the start time of the animation.
CanMakeUnchangeable	True if this Changeable can be made unchangeable. Inherited from Changeable.
CurrentRepeat	Gets the number of the current iteration of the animation.
CurrentTime	Gets the current time value of the animation.
Deceleration	Gets or sets a value that represents the fraction of the simple duration spent in the deceleration phase.
Duration	Gets or sets the length of time the animation takes to complete a single forward iteration, also known as the simple duration of an animation.
End	Gets or sets the maximum end time of the animation.
EndSync	Gets or sets a TimeEndSync enumeration that specifies how the implicit duration of an animation is determined. This property is only used if the Duration property is not explicitly set.
Fill	Gets or sets a value that specifies the state of an object when its animation ends.
FillDefault	Gets or sets a value that indicates the default value of the Fill property of the current animation and its child timelines.
IsChangeable	Gets a Boolean that indicates whether the object is currently modifiable. Inherited from Changeable.
IsChanging	Gets a value that indicates whether the animation is active.
IsEnabled	
IsForwardProgressing	Gets a value that indicates whether the animation is progressing from past to future.
IsOverridingBaseValue	Gets a value that indicates whether the animation is active or in a fill period.
IsPaused	Gets a value that indicates whether the animation is active and paused.
IsReversed	Gets a value that indicates whether the animation is currently moving in the opposite direction of its parent timeline.
ParentTimeline	Gets or sets the default parent timeline of the animation.
Progress	Gets a number from 0 to 1 that indicates the fraction of the simple duration that has elapsed.
RepeatCount	Gets or sets the number of times an animation should repeat.
RepeatDuration	Gets or sets the total length of time the animation should play. If this value is greater than the simple duration of the animation, it will repeat itself for the length of time specified by this property.
Restart	Gets or sets the animation's behavior when it is told to restart—that is, how the animation behaves when a second begin time is reached.
RestartDefault	Gets or sets the default value of the Restart property of the current